C 12

```
   SSSSSSSSSSSS  MMM         MMM     GGGGGGGGGGGG  RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
   SSSSSSSSSSSS  MMM         MMM     GGGGGGGGGGGG  RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
   SSSSSSSSSSSS  MMM         MMM     GGGGGGGGGGGG  RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
SSS              MMMMMM   MMMMMM  GGG              RRR         RRR        TTT        LLL
SSS              MMMMMM   MMMMMM  GGG              RRR         RRR        TTT        LLL
SSS              MMMMMM   MMMMMM  GGG              RRR         RRR        TTT        LLL
SSS              MMM   MMM   MMM  GGG              RRR         RRR        TTT        LLL
SSS              MMM   MMM   MMM  GGG              RRR         RRR        TTT        LLL
SSS              MMM   MMM   MMM  GGG              RRR         RRR        TTT        LLL
   SSSSSSSSS     MMM         MMM  GGG              RRRRRRRRRRRR           TTT        LLL
   SSSSSSSSS     MMM         MMM  GGG              RRRRRRRRRRRR           TTT        LLL
   SSSSSSSSS     MMM         MMM  GGG              RRRRRRRRRRRR           TTT        LLL
            SSS  MMM         MMM  GGG   GGGGGGGGG  RRR   RRR              TTT        LLL
            SSS  MMM         MMM  GGG   GGGGGGGGG  RRR   RRR              TTT        LLL
            SSS  MMM         MMM  GGG   GGGGGGGGG  RRR   RRR              TTT        LLL
            SSS  MMM         MMM  GGG         GGG  RRR      RRR           TTT        LLL
            SSS  MMM         MMM  GGG         GGG  RRR      RRR           TTT        LLL
            SSS  MMM         MMM  GGG         GGG  RRR      RRR           TTT        LLL
SSSSSSSSSSSS     MMM         MMM     GGGGGGGGG     RRR         RRR        TTT        LLLLLLLLLLLLLLL
SSSSSSSSSSSS     MMM         MMM     GGGGGGGGG     RRR         RRR        TTT        LLLLLLLLLLLLLLL
SSSSSSSSSSSS     MMM         MMM     GGGGGGGGG     RRR         RRR        TTT        LLLLLLLLLLLLLLL
```

```
_$2

Val
---
001
001
001
001
001
001
001
001
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
```

```
**FILE**ID**SMGDISLIN

  SSSSSSSS   MM      MM     GGGGGGGG   DDDDDDDD       IIIIII       SSSSSSSS   LL             IIIIII     NN      NN
  SSSSSSSS   MM      MM     GGGGGGGG   DDDDDDDD       IIIIII       SSSSSSSS   LL             IIIIII     NN      NN
SS           MMMM  MMMM   GG           DD      DD       II       SS           LL               II       NN      NN
SS           MMMM  MMMM   GG           DD      DD       II       SS           LL               II       NN      NN
SS           MM  MM  MM   GG           DD      DD       II       SS           LL               II       NNNN    NN
SS           MM  MM  MM   GG           DD      DD       II       SS           LL               II       NNNN    NN
  SSSSSS     MM      MM   GG           DD      DD       II         SSSSSS     LL               II       NN  NN  NN
  SSSSSS     MM      MM   GG           DD      DD       II         SSSSSS     LL               II       NN  NN  NN
        SS   MM      MM   GG  GGGGGG   DD      DD       II               SS   LL               II       NN    NNNN
        SS   MM      MM   GG  GGGGGG   DD      DD       II               SS   LL               II       NN    NNNN
        SS   MM      MM   GG      GG   DD      DD       II               SS   LL               II       NN      NN   ....
        SS   MM      MM   GG      GG   DD      DD       II               SS   LL               II       NN      NN   ....
SSSSSSSS     MM      MM     GGGGGG     DDDDDDDD       IIIIII     SSSSSSSS     LLLLLLLLLL     IIIIII     NN      NN   ....
SSSSSSSS     MM      MM     GGGGGG     DDDDDDDD       IIIIII     SSSSSSSS     LLLLLLLLLL     IIIIII     NN      NN   ....

LL             IIIIII       SSSSSSSS
LL             IIIIII       SSSSSSSS
LL               II       SS
LL               II       SS
LL               II       SS
LL               II       SS
LL               II         SSSSSS
LL               II         SSSSSS
LL               II               SS
LL               II               SS
LL               II               SS
LL               II               SS
LLLLLLLLLL     IIIIII     SSSSSSSS
LLLLLLLLLL     IIIIII     SSSSSSSS
```

```
0 %TITLE 'SMG$DISPLAY_LINKS - Virtual Display Linkages'
0 MODULE SMG$DISPLAY_LINKS (
0                 IDENT = '1-096' ! File: SMGDISLIN.B32 Edit: STAN1096
0                 ) =
1 BEGIN
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1 !
1
1 !++
1 ! FACILITY:      Screen Management
1 !
1 ! ABSTRACT:
1 !       The procedures in this module are conerned only with the
1 ! allocation/deallocation of virtual displays, and with the pasting/
1 ! unpasting of these virtual displays to pasteboards.  The are not
1 ! concerned with their contents or output.
1 !
1 !       For the procedures which maintain and update the contents of
1 ! virtual displays, see the module SMG$DISPLAY_CHANGE.
1 !
1 !       For the procedures which actually do output from these virtual
1 ! displays, see the module SMG$DISPLAY_OUTPUT.
1 !
1 !       For procedures that support input operations, see the module
1 ! SMG$DISPLAY_INPUT.
1 !
1 ! ENVIRONMENT:  User mode, Shared library routines.
1 !
1 ! AUTHOR: R. Reichert, CREATION DATE: 26-Jan-1983
1 !
1 ! MODIFIED BY:
1 !
1 ! 1-096 - Don't allow paste or unpaste if display is batched.
1 !         STAN 27-Jun-1984.
1 ! 1-095 - Use symbolic names SMG$K_TOP, etc. in SMG$LABEL_BORDER.
```

```
;       1 !             Change error messages by SMG$SET_DISPLAY_SCROLLING_REGION.
;       1 !             STAN 3-Jun-1984.
;       1 ! 1-094 - Fix bug re borders occluding other borders. STAN 7-May-1984.
;       1 ! 1-001 - Original.  Skeleton for future code.  RKR 26-Jan-1983
;       1 !--
```

```
1 %SBTTL 'Declarations'
1 !
1 ! SWITCHES:
1 !
1
1 !
1 ! LINKAGES:
1 !
1 !      NONE
1 !
1 ! INCLUDE FILES
1 !
1
1 REQUIRE 'RTLIN:SMGPROLOG';                    ! defines psects, macros, tcb,
1                                               !  wcb, & terminal symbols
1
1 REQUIRE 'RTLIN:STRLNK';                       ! Linkages to string JSB
1
1 !
1 ! TABLE OF CONTENTS:
1 !
1
1 FORWARD ROUTINE
1
1 ! Public entry points
1
1     SMG$CHANGE_PBD_CHARACTERISTICS,           ! Change characteristics of
1                                               ! physical terminal
1
1     SMG$CHANGE_VIRTUAL_DISPLAY,               ! Change characteristics of
1                                               ! existing virtual display
1
1     SMG$CHECK_FOR_OCCLUSION,                  ! Check to see if a virtual
1                                               ! display is occluded.
1
1     SMG$CREATE_PASTEBOARD,                    ! Create pasteboard
1
1     SMG$CREATE_VIRTUAL_DISPLAY,               ! Create virtual display
1
1     SMG$DELETE_PASTEBOARD,                    ! Get rid of pasteboard, terminate
1                                               ! all operations on this display
1
1     SMG$DELETE_VIRTUAL_DISPLAY,               ! Delete virtual display
1
1     SMG$GET_DISPLAY_ATTR,                     ! Return current attributes of
1                                               ! virtual display
1
1     SMG$LABEL_BORDER,                         ! Supply label for border
1
1     SMG$MOVE_VIRTUAL_DISPLAY,                 ! Move position of virtual
1                                               ! display on pasteboard
1
1     SMG$PASTE_VIRTUAL_DISPLAY,                ! Paste virtual display to
1                                               ! pasteboard
1
1     SMG$POP_VIRTUAL_DISPLAY,                  ! Pop off (and delete) all
1                                               ! virtual displays from given
```

```
1                                               ! to top of pasting stack.
1
1       SMG$REPASTE_VIRTUAL_DISPLAY,            ! Repaste virtual display to
1                                               ! pasteboard in new position
1
1       SMG$RESTORE_PHYSICAL_SCREEN,            ! Restore screen to where it
1                                               ! was after non-SMG user
1                                               ! munged it.
1
1       SMG$SAVE_PHYSICAL_SCREEN,               ! Save physical screen before
1                                               ! non-SMG user mungs its up.
1
1       SMG$SET_DISPLAY_SCROLL_REGION,          ! Set the scrolling region in
1                                               ! a virtual display
1
1       SMG$UNPASTE_VIRTUAL_DISPLAY,            ! Unpaste virtual display from
1                                               ! pasteboard.
1
1 ! Private entry points
1
1       SMG$$CALC_PASTE_TRANSF,                 ! Calculate pasting
1                                               ! transformation constants.
1
1       SMG$$CHECK_OCCLUSION,                   ! Check current complement of
1                                               ! pasted virtual displays to
1                                               ! see who is occluded.
1
1       SMG$$CHECK_OCCLUSION_FIRST,             ! Check occlusion caused by
1                                               ! highest pasted virtual display.
1
1       SMG$$CREATE_PASTEBOARD,                 ! Create pasteboard
1
1       SMG$$CREATE_VIRTUAL_DISPLAY,            ! Inner-most Create Virtual
1                                               ! Display routine
1
1       SMG$$CREATE_WCB,                        ! Create WCB and its buffers
1
1       SMG$$DEALLOCATE_WCB,                    ! Get rid of WCB and its buffers.
1
1       SMG$$DUPL_VIRTUAL_DISPLAY,              ! Duplicate a virtual display
1
1       SMG$$LOCATE_PP,                         ! Locate PP which matches a
1                                               ! DCB and a PBCB.
1
1       SMG$$PASTE_VIRTUAL_DISPLAY,             ! Inner-most Paste Virtual
1                                               ! Display routine.
1
1       SMG$$RECALC_PP_FIELDS,                  ! Recalculate pasting packet
1                                               ! fields after virtual display
1                                               ! batching ceases.
1       SMG$$UNPASTE_VIRTUAL_DISPLAY;           ! Inner-most Unpaste Virtual
1                                               ! Display routine.
1
1
1                                               ! routines.
1
1 !
```

```
1 ! EXTERNAL REFERENCES
1 !
1 EXTERNAL ROUTINE
1       LIB$ANALYZE_SDESC_R2 : LIB$ANALYZE_SDESC_JSB_LINK,
1                                   ! Get length and address of a string
1
1       LIB$FREE_VM,                ! Deallocate heap storage
1
1       LIB$FREE_EF,                ! Free an event flag
1
1       LIB$GET_EF,                 ! Get an event flag
1
1       LIB$GET_VM,                 ! Allocate heap storage
1
1       LIB$SCOPY_DXDX,             ! String copy by descriptor
1
1       LIB$SFREE1_DD,              ! Free a dynamic string
1
1       SMG$$BEGIN_PASTEBOARD_UPDATE_R1 : SMG$$BEGIN_PBD_UPDATE$LNK,
1                                   ! Increase buffering level by 1
1
1       SMG$$END_PASTEBOARD_UPDATE_R2   : SMG$$END_PBD_UPDATE$LNK,
1                                   ! Decrease buffering level by 1
1
1       SMG$$ERASE_PASTEBOARD,      ! Erase the physical screen
1
1       SMG$$CHECK_FOR_OUTPUT_DCB,  ! Force output if now is the time
1
1       SMG$$CHECK_FOR_OUTPUT_PBCB, ! Force output
1
1       SMG$$FILL_WINDOW_BUFFER,    ! Move stuff from virt. display to
1                                   ! pasteboard buffer and output.
1
1       SMG$$FIND_MIN_CURSOR_POS,   ! Set cursor on physical screen
1       SMG$$FLUSH_BUFFER,          ! Flush output buffer
1
1       SMG$$FORCE_SCROLL_REG,      ! Force scrolling region on screen.
1
1       SMG$$OUTPUT,                ! Output a string to terminal
1
1       SMG$$OCCLUDE,               ! Check for how two rectangular areas
1                                   ! overlap.
1
1       SMG$$PBCB_EXIT_HANDLER,     ! Output exit handler
1
1       SMG$$SETUP_TERMINAL_TYPE;   ! Get device characteristics
1
1 EXTERNAL LITERAL
1
1       LIB$_EF_ALRFRE,     ! Event flag already free
1       SMG$_BATWAS_ON,     ! Batching was enabled
1       SMG$_FATERRLIB,     ! Fatal error in library
1       SMG$_INVARG,        ! Invalid argument
1       SMG$_ILLBATFNC,     ! Operation not legal to batched display
1       SMG$_INVDIS_ID,     ! Invalid virtual display id
1       SMG$_INVPAS_ID,     ! Invalid pasteboard id
1       SMG$_INVROW,        ! Invalid row
```

```
;  235          0495  1      SMG$_NOTPASTED,        ! Given display is not pasted to given
;  236          0496  1                             ! pasteboard
;  237          0497  1      SMG$_PASALREXI,        ! Pasteboard already exists for this device
;  238          0498  1      SMG$_TOOMANDIS,        ! Too many virtual displays requested
;  239          0499  1      SMG$_TOOMANPAS,        ! Too many pasteboards requested
;  240          0500  1      SMG$_WRONUMARG;        ! Wrong number of arguments
```

SMG
1-0

```
!+
! Pasteboard Directory (PBD)
! --------------------------
! This data structure resides in OWN storage.  It is the primary vehicle
! for getting from a pasteboard id to the associated pasteboard control
! block.
!-

GLOBAL
    PBD_L_COUNT : INITIAL (0),  ! No. of pasteboards we currently know
                                ! about.

    PBD_A_PBCB : VECTOR  [PBD_K_MAX_PB, LONG]
                 INITIAL (REP PBD_K_MAX_PB OF (0)),
                        ! List of pasteboard addresses. Indexed by
                        ! pasteboard id (PID) to find address of
                        ! corresponding PBCB.

    PBD_V_PB_AVAIL : BITVECTOR [PBD_K_MAX_PB]
                     INITIAL ( BYTE (REP ((PBD_K_MAX_PB+7)/8) OF (0)));
                        ! This is a bit-vector of pasteboard id's
                        ! still available.  The next available number
                        ! is found by doing a FFC instruction to find
                        ! first bit which is a 0.  The bit position so
                        ! computed is the next available PID.  The
                        ! bit found is set to 1 to mark it as in use.
                        ! (Presumably, a check has already been made to
                        ! insure that PBD_L_COUNT is LSS PBD_K_MAX_PB.)

! Some constants needed by reference for FFC instruction
OWN
    ZERO                : INITIAL ( 0 ),
    PBD_K_MAX_PB_BY_REF : INITIAL ( PBD_K_MAX_PB );
```

```
1 %SBTTL 'SMG$CHANGE_PBD_CHARACTERISTICS'
1 GLOBAL ROUTINE SMG$CHANGE_PBD_CHARACTERISTICS
1         (PBID,
1          P_DESIRED_WIDTH,
1          P_RESULTING_WIDTH,
1          P_DESIRED_HEIGHT,
1          P_RESULTING_HEIGHT,
1          P_DESIRED_BACKGROUND_COLOR,
1          P_RESULTING_BACKGROUND_COLOR
1         )=
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine lets you change the physical dimensions
1 !       of a pasteboard.  It also lets you change the background color.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       ret_status.wlc.v = SMG$CHANGE_PBD_CHARACTERISTICS
1 !                                       ( PBID.rl.r
1 !                                       [,DESIRED_WIDTH.rl.r]
1 !                                       [,RESULTING_WIDTH.wl.r]
1 !                                       [,DESIRED_HEIGHT.rl.r]
1 !                                       [,RESULTING_HEIGHT.wl.r]
1 !                                       [,DESIRED_BACKGROUND_COLOR.rl.r]
1 !                                       [,RESULTING_BACKGROUND_COLOR.wl.r]
1 !                                       )
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       PBID.rl.r               Pasteboard id of pasteboard.
1 !
1 !       DESIRED_WIDTH.rl.r      New width desired for pasteboard.
1 !                               If omitted, the width is not changed.
1 !
1 !       RESULTING_WIDTH.wl.r    Physical width that resulted.  This may
1 !                               be larger than the width requested if the
1 !                               terminal width couldn't be set exactly to
1 !                               the desired width.  This may be smaller
1 !                               than the width requested if the terminal
1 !                               width couldn't be set that wide.
1 !                               In this case, the terminal was set to
1 !                               it's maximum width.
1 !
1 !                               Example:        (for VT100)
1 !
1 !                               Width Desired   Width resulting
1 !
1 !                               60              80
1 !                               110             132
1 !                               150             132
1 !
1 !                               If desired width was omitted, this
1 !                               argument receives the current pasteboard
1 !                               width.
1 !                               To find out what the pasteboard width is
```

```
1 !                                   (as opposed to the terminal width),
1 !                                   the caller should take the minimum
1 !                                   of his desired width and the resulting width.
1 !
1 !     DESIRED_HEIGHT.rl.r           New height desired for pasteboard.
1 !                                   If omitted, the height is not changed.
1 !
1 !     RESULTING_HEIGHT.wl.r         Physical height that resulted.  This may
1 !                                   be larger than the height requested if the
1 !                                   terminal height couldn't be set exactly to
1 !                                   the desired height.  This may be smaller
1 !                                   than the height requested if the terminal
1 !                                   height couldn't be set that high.
1 !                                   In this case, the terminal was set to
1 !                                   it's maximum height.
1 !
1 !                                   Example:        (for VT100)
  !
  !                                   Height Desired  Height resulting
1 !
1 !                                   15              24
1 !
1 !                                   35              24
1 !
1 !                                   To find out what the pasteboard height is
1 !                                   (as opposed to the terminal height),
1 !                                   the caller should take the minimum
1 !                                   of his desired height and the resulting height.
1 !
1 !     DESIRED_BACKGROUND_COLOR.rl.r       Symbolic name for the background
1 !                                         color wanted.  For example,
1 !                                         SMG$C_COLOR_WHITE.  These symbols
1 !                                         are defined in SMGDEF.SDL.
1 !                                         If omitted, the background color
1 !                                         is not changed.
1 !
1 !     RESULTING_BACKGROUND_COLOR.wl.r Receives the actual background color
1 !                                         that was chosen.  If the terminal
1 !                                         does not support the exact color
1 !                                         desired, the nearest approximation
1 !                                         will be chosen.  This is determined
1 !                                         by comparing the frequency of the
1 !                                         desired light wave against the
1 !                                         available frequencies.  For more
1 !                                         information about colorimetry,
1 !                                         consult National Bureau of Standards
1 !                                         Circular 553, The ISCC-NBS method of
1 !                                         designating Colors.
1 !
1 !                                         Example: (VT100)
1 !
1 !                                         Color desired   Resulting Color
1 !
1 !                                         yellowish pink  white
1 !
1 !                                         navy blue       black
1 !
```

```
1 !                                                  If the desired color is omitted,
1 !                                                  the value of this variable is not
1 !                                                  affected.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !       SS$_NORMAL        Normal successful completion
1 !       SMG$_WRONUMARG    Wrong number of arguments.
1 !       SMG$_PBDIN_USE    Can't change characteristics while buffering is on
1 !       SMG$_INVWIDARG    Invalid width of 0 desired
1 !       SMG$_INVPAGARG    Invalid height of 0 desired
1 !       SMG$_INVCOLARG    Unknown background color specified
1 !
1 ! SIDE EFFECTS:
1 !
1 !       Physical width and background color of terminal may change.
1 !--
```

```
;      2 BEGIN
;      2
;      2 EXTERNAL ROUTINE
;      2
;      2         SMG$$CHECK_FOR_OUTPUT_PBCB,
;      2         SMG$$CALC_PASTE_TRANSF,
;      2         SMG$$CREATE_WCB,
;      2         SMG$$DEALLOCATE_WCB,
;      2         SMG$$ERASE_PASTEBOARD,
;      2         SMG$$OUTPUT;
;      2
;      2 EXTERNAL LITERAL
;      2
;      2         SMG$_INVWIDARG,          ! width=0
;      2         SMG$_INVPAGARG,          ! HEIGHT=0
;      2         SMG$_INVCOLARG,          ! unknown color
;      2         SMG$_PBDIN_USE;          ! pasteboard was batched
;      2
;      2 BUILTIN
;      2
;      2         NULLPARAMETER;
;      2
;      2 LOCAL
;      2
;      2         STATUS,                          ! Status of subroutine calls
;      2         PASTING_PACKET_PANIC,            ! TRUE if we must adjust pasting packets
;      2         CURR_PP  : REF $PP_DECL,         ! Pasting packet pointer
;      2
;      2         PBCB     : REF $PBCB_DECL;       ! Address of pasteboard control block
```

```
2 $SMG$VALIDATE_ARGCOUNT (1, 7);  ! Test for right no. of args
2
2 $SMG$GET_PBCB (.PBID,PBCB);     ! Get address of PBCB
2
2 PASTING_PACKET_PANIC=0;
2
2 !+
2 ! If a desired width is specified, get it now.
2 !-
2
2 IF NOT NULLPARAMETER(P_DESIRED_WIDTH)
3  THEN  BEGIN   ! Change pasteboard width
3        BIND    DESIRED_WIDTH=.P_DESIRED_WIDTH;
3 !(a)   LOCAL   CURRENT_MAX, DESIRED_MAX;
3        LOCAL   PREVIOUS_WIDTH;
3        LOCAL   RESULTANT_WIDTH;
3
3        IF .DESIRED_WIDTH EQL 0
3          THEN  RETURN  SMG$_INVWIDARG;
3
3        !+
3        ! Determine the physical setting of the terminal by rounding
3        ! up to 80 or 132 as necessary.  Do the same for the desired
3        ! width.  Compare these two numbers to see if we must change
3        ! the width.  This algorithm will have to change if we ever
3        ! support terminals with widths other than 80 and 132.
3        !-
3
3        IF .PBCB[PBCB_L_BATCH_LEVEL] NEQ 0
3          THEN  RETURN  SMG$_PBDIN_USE;
3 !(a)   IF .PBCB[PBCB_W_WIDTH] LEQU 80
3 !(a)     THEN  CURRENT_MAX=80
3 !(a)     ELSE  CURRENT_MAX=132;
3 !(a)   IF .DESIRED_WIDTH LEQU 80
3 !(a)     THEN  DESIRED_MAX=80
3 !(a)     ELSE  DESIRED_MAX=132;
3
3 !(a)   !+
3 !(a)   ! If the desired max is the same as the current max,
3 !(a)   ! then no escape sequence need be sent to the terminal.
3 !(a)   ! Just adjust our internal width in the PBCB.
3 !(a)   !-
3
3 !(a)   IF .DESIRED_MAX NEQ .CURRENT_MAX
3 !(a)     THEN
3
3 !+
3 !
3 !      Note: (a)
3 !
3 !      The lines marked !(a) could be added back in
3 !      if you want to avoid outputting the escape sequence
3 !      to change the terminal width if it isn't necessary.
3 !      However, that will mean the screen doesn't physically
3 !      blank and so extra code would have to be written to
3 !      blank the right part of a screen when changing width
3 !      (say) from 70 to 50 columns.
```

```
3 !-
3
4                         BEGIN   ! Change physical width
4
4                         LOCAL
4
4                                 NORMAL_WIDTH,
4                                 WIDE_WIDTH;
4
4                         !+
4                         ! First, clear the screen.
4                         !-
4
4                         $SMG$GET_TERM_DATA(ERASE_WHOLE_DISPLAY);
4                         IF .PBCB[PBCB_L_CAP_LENGTH] NEQ 0
5                           THEN  BEGIN
5                                 STATUS = SMG$$OUTPUT(.PBCB,.PBCB[PBCB_L_CAP_LENGTH],
5                                                          .PBCB[PBCB_A_CAP_BUFFER]);
5                                 IF NOT .STATUS THEN RETURN .STATUS
4                                 END;
4
4                         !+
4                         ! Second, get the normal size.
4                         !-
4
4                         $SMG$GET_TERM_DATA(COLUMNS);
4                         IF .PBCB[PBCB_L_CAP_LENGTH] NEQ 0
5                           THEN  BEGIN
5                                 BIND RESULT=.PBCB[PBCB_A_CAP_BUFFER];
5                                 STATUS = SMG$$OUTPUT(.PBCB,.PBCB[PBCB_L_CAP_LENGTH],
5                                                          .PBCB[PBCB_A_CAP_BUFFER]);
5                                 IF NOT .STATUS THEN RETURN .STATUS;
5                                 NORMAL_WIDTH=.RESULT
5                                 END
4                           ELSE  NORMAL_WIDTH=80;
4
4                         !+
4                         ! Third, get the wide size.
4                         !-
4
4                         $SMG$GET_TERM_DATA(WIDTH_WIDE);
4                         IF .PBCB[PBCB_L_CAP_LENGTH] NEQ 0
5                           THEN  BEGIN
5                                 BIND RESULT=.PBCB[PBCB_A_CAP_BUFFER];
5                                 STATUS = SMG$$OUTPUT(.PBCB,.PBCB[PBCB_L_CAP_LENGTH],
5                                                          .PBCB[PBCB_A_CAP_BUFFER]);
5                                 IF NOT .STATUS THEN RETURN .STATUS;
5                                 WIDE_WIDTH=.RESULT
5                                 END
4                           ELSE  WIDE_WIDTH=80;
4
4                         !+
4                         ! Decide which sequence to send.
4                         !-
4
4                         IF .DESIRED_WIDTH LEQ .NORMAL_WIDTH
5                           THEN  BEGIN
```

```
5                               $SMG$GET_TERM_DATA(WIDTH_NARROW);
5                               RESULTANT_WIDTH=.NORMAL_WIDTH
5                               END
5                         ELSE  BEGIN
5                               $SMG$GET_TERM_DATA(WIDTH_WIDE);
5                               RESULTANT_WIDTH=.WIDE_WIDTH
4                               END;
4
4                       IF .PBCB[PBCB_L_CAP_LENGTH] NEQ 0
5                         THEN  BEGIN
5                               STATUS = SMG$$OUTPUT(.PBCB,.PBCB[PBCB_L_CAP_LENGTH],
5                                                   .PBCB[PBCB_A_CAP_BUFFER]);
5                               IF NOT .STATUS THEN RETURN .STATUS;
4                               END;
4
4                       !+
4                       ! If we asked for something smaller than the terminal
4                       ! could handle (like a width of 60 on an 80-column terminal)
4                       ! then we will software simulate the smaller width.
4                       !-
4
4                       RESULTANT_WIDTH=MINU(.RESULTANT_WIDTH,.DESIRED_WIDTH);
4
3                       END;    ! Change physical width
3
3               !+
3               ! Should we go back to the old scheme whereby we
3               ! output the escape sequence only if the max width has
3               ! changed, then we need the following line:
3
3               !(a)  ELSE      RESULTANT_WIDTH=.DESIRED_WIDTH;
3               !-
3
3               !+
3               ! Save away new pasteboard width in the PBCB.
3               !-
3
3               PREVIOUS_WIDTH=.PBCB[PBCB_W_WIDTH];
3               PBCB[PBCB_W_WIDTH]=.RESULTANT_WIDTH;
3
3               !+
3               ! If the width changed, we must recalculate all the pasting
3               ! packet parameters pronto. Make a note.
3               !-
3
3 !             IF .PREVIOUS_WIDTH NEQ .RESULTANT_WIDTH
3 !               THEN  PASTING_PACKET_PANIC=1;
3               PASTING_PACKET_PANIC=1;
3
3               !+
3               ! At some point in the future, we might want to tell VMS
3               ! about this new width.  If so, we would add that code here.
3               ! There is probably no need to do that since we will restore
3               ! the original width when we delete this pasteboard.
3               !-
3
2               END;    ! Change pasteboard width
```

```
2
2 !+
2 ! If the user wants the pasteboard width, give it to him now.
2 !-
2
2 IF NOT NULLPARAMETER(P_RESULTING_WIDTH)
3   THEN  BEGIN    ! Return pasteboard width
3         BIND     RESULTING_WIDTH = .P_RESULTING_WIDTH;
3         RESULTING_WIDTH=.PBCB[PBCB_W_WIDTH]
2         END;     ! Return pasteboard width
2
2 !+
2 ! If the user wants to change his height, do that now.
2 ! If he specifies an illegal height, that's his problem;
2 ! we don't know what sort of funny terminal he might have.
2 ! This code will have to change if we ever support terminals
2 ! that can change height by sending them escape sequences.
2 !-
2
2 IF NOT NULLPARAMETER(P_DESIRED_HEIGHT)
3   THEN  BEGIN    ! Change pasteboard height
3         BIND     DESIRED_HEIGHT = .P_DESIRED_HEIGHT;
3         IF .PBCB[PBCB_L_BATCH_LEVEL] NEQ 0
3           THEN  RETURN  SMG$_PBDIN_USE;
3         IF .DESIRED_HEIGHT EQL 0
3           THEN  RETURN  SMG$_INVPAGARG;
3         IF .PBCB[PBCB_B_ROWS] NEQ .DESIRED_HEIGHT
4           THEN  BEGIN
4                 !+
4                 ! Blank screen if we are making screen smaller,
4                 ! so as to get rid of items after the bottom of
4                 ! the pasteboard.
4                 !-
4                 IF MINU(24,.DESIRED_HEIGHT) LSSU .PBCB[PBCB_B_ROWS]
5                   THEN  BEGIN
5                         STATUS=SMG$$ERASE_PASTEBOARD(.PBCB);
5                         IF NOT .STATUS THEN RETURN .STATUS
4                         END;
4                 !+
4                 ! All existing terminals have a maximum height of 24.
4                 !-
4                 PBCB[PBCB_B_ROWS]=MINU(24,.DESIRED_HEIGHT);
4                 PASTING_PACKET_PANIC=1
4                 END
2         END;     ! Change pasteboard height
2
2 !+
2 ! If the user wants the pasteboard height, give it to him now.
2 !-
2
2 IF NOT NULLPARAMETER(P_RESULTING_HEIGHT)
3   THEN  BEGIN    ! Return pasteboard height
3         BIND     RESULTING_HEIGHT=.P_RESULTING_HEIGHT;
3         RESULTING_HEIGHT=.PBCB[PBCB_B_ROWS]
2         END;     ! Return pasteboard height
2
2 !+
2 ! If we changed either the width or height of the pasteboard,
2 ! then we must go adjust all the pasting packets now.
2 ! We must also reallocate and reshape the buffers in the WCB.
2 !-
2
2 IF .PASTING_PACKET_PANIC
3   THEN  BEGIN    ! Update all pasting packets
3         LOCAL CURR_PP    : REF $PP_DECL;
3         !+
3         ! Deallocate the old WCB.
3         !-
3         STATUS=SMG$$DEALLOCATE_WCB(.PBCB[PBCB_A_WCB]);
3         IF NOT .STATUS THEN RETURN .STATUS;
3         !+
3         ! Allocate a new WCB.
3         !-
3         STATUS=SMG$$CREATE_WCB( %REF(.PBCB[PBCB_B_ROWS]),
3                                 %REF(.PBCB[PBCB_W_WIDTH]),
3                                 PBCB[PBCB_A_WCB]);
3         IF NOT .STATUS THEN RETURN .STATUS;
3         !+
3         ! Walk chain of DCB's for all displays currently pasted
3         ! to this pasteboard, and go update their pasting packet.
3         ! Start with first packet.
3         !-
3         CURR_PP=.PBCB[PBCB_A_PP_NEXT];
3         WHILE .CURR_PP NEQ PBCB[PBCB_A_PP_NEXT] DO
4                 BEGIN    ! Update a pasting packet
4                 LOCAL
4                     PP_BASE      : REF $PP_DECL;          ! Base addr of this PP
4
4                 PP_BASE = .CURR_PP - PP_PBCB_QUEUE_OFFSET;  ! Since queue header
4                                                             ! not at top of
4                                                             ! structure.
4                 STATUS=SMG$$CALC_PASTE_TRANSF(.PP_BASE);
4                 IF NOT .STATUS THEN RETURN .STATUS;
4                 CURR_PP = .PP_BASE [PP_A_NEXT_PBCB] ! Step to next PP
3                 END;     ! Update a pasting packet
3
3         !+
3         ! Force an update.
3         !-
3
3         STATUS=SMG$$CHECK_FOR_OUTPUT_PBCB(.PBCB);
3         IF NOT .STATUS THEN SIGNAL(.STATUS);
3
2         END;     ! Update all pasting packets
2
2 !+
2 ! If a new background color is desired, go do that now.
2 !-
2
2 IF NOT NULLPARAMETER(P_DESIRED_BACKGROUND_COLOR)
3   THEN  BEGIN    ! Change background color
3         BIND    DESIRED_COLOR=.P_DESIRED_BACKGROUND_COLOR;
3         BIND    RESULTING_COLOR=PBCB[PBCB_B_BACKGROUND_COLOR];
3
```

```
;   731          0987  3            IF .DESIRED_COLOR EQL SMG$C_COLOR_WHITE
;   732          0988  4              THEN  $SMG$GET_TERM_DATA(LIGHT_SCREEN)
;   733          0989  3              ELSE  $SMG$GET_TERM_DATA(DARK_SCREEN);
;   734          0990  3
;   735          0991  3            IF .PBCB[PBCB_L_CAP_LENGTH] NEQ 0
;   736          0992  4              THEN  BEGIN
;   737          0993  4                    STATUS = SMG$$OUTPUT(.PBCB,.PBCB[PBCB_L_CAP_LENGTH],
;   738          0994  4                                          .PBCB[PBCB_A_CAP_BUFFER]);
;   739          0995  4                    IF NOT .STATUS THEN RETURN .STATUS
;   740          0996  3                    END;
;   741          0997  3
;   742          0998  3            RESULTING_COLOR=.DESIRED_COLOR;
;   743          0999  3
;   744          1000  2            END;     ! Change terminal coror
;   745          1001  2
;   746          1002  2 !+
;   747          1003  2 ! If the user wants the new background color, give it to him now.
;   748          1004  2 !-
;   749          1005  2
;   750          1006  2 IF NOT NULLPARAMETER(P_RESULTING_BACKGROUND_COLOR)
;   751          1007  3   THEN  BEGIN    ! Return background color
;   752          1008  3         BIND     RESULTING_COLOR=.P_RESULTING_BACKGROUND_COLOR;
;   753          1009  3         RESULTING_COLOR=.PBCB[PBCB_B_BACKGROUND_COLOR]
;   754          1010  2         END;     ! Return background color
;   755          1011  2
;   756          1012  2 RETURN  SS$_NORMAL
;   757          1013  2
;   758          1014  1 END;                           ! Routine SMG$CHANGE_PBD_CHARACTERISTICS


                                                                      .TITLE  SMG$DISPLAY_LINKS SMG$DISPLAY_LINKS - Virtual D
;                                                                                                 isplay Linkages
                                                                      .IDENT  \1-096\

                                                                      .PSECT  _SMG$DATA,NOEXE,  PIC,2

                                                  00000000  00000 PBD_L_COUNT::
                                                                      .LONG   0                                              ;
                                                  00000000# 00004 PBD_A_PBCB::
                                                                      .LONG   0[16]                                          ;
                                                        00# 00044 PBD_V_PB_AVAIL::
                                                                      .BYTE   0[2]                                           ;
                                                            00046     .BLKB   2
                                                  00000000  00048 ZERO:  .LONG   0                                           ;
                                                  00000010  0004C PBD_K_MAX_PB_BY_REF:
                                                                      .LONG   16                                             ;

                                                                      .EXTRN  LIB$ANALYZE_SDESC_R2
                                                                      .EXTRN  LIB$FREE_VM, LIB$FREE_EF
                                                                      .EXTRN  LIB$GET_EF, LIB$GET_VM
                                                                      .EXTRN  LIB$SCOPY_DXDX, LIB$SFREE1_DD
                                                                      .EXTRN  SMG$$BEGIN_PASTEBOARD_UPDATE_R1
                                                                      .EXTRN  SMG$$END_PASTEBOARD_UPDATE_R2
                                                                      .EXTRN  SMG$$ERASE_PASTEBOARD
                                                                      .EXTRN  SMG$$CHECK_FOR_OUTPUT_DCB
                                                                      .EXTRN  SMG$$CHECK_FOR_OUTPUT_PBCB
                                                                      .EXTRN  SMG$$FILL_WINDOW_BUFFER
                                                  .EXTRN  SMG$$FIND_MIN_CURSOR_POS
                                                  .EXTRN  SMG$$FLUSH_BUFFER
                                                  .EXTRN  SMG$$FORCE_SCROLL_REG
                                                  .EXTRN  SMG$$OUTPUT, SMG$$OCCLUDE
                                                  .EXTRN  SMG$$PBCB_EXIT_HANDLER
                                                  .EXTRN  SMG$$SETUP_TERMINAL_TYPE
                                                  .EXTRN  LIB$_EF_ALRFRE, SMG$_BATWAS_ON
                                                  .EXTRN  SMG$_FATERRLIB, SMG$_INVARG
                                                  .EXTRN  SMG$_ILLBATFNC, SMG$_INVDIS_ID
                                                  .EXTRN  SMG$_INVPAS_ID, SMG$_INVROW
                                                  .EXTRN  SMG$_NOTPASTED, SMG$_PASALREXI
                                                  .EXTRN  SMG$_TOOMANDIS, SMG$_TOOMANPAS
                                                  .EXTRN  SMG$_WRONUMARG, SMG$$CALC_PASTE_TRANSF
                                                  .EXTRN  SMG$$CREATE_WCB
                                                  .EXTRN  SMG$$DEALLOCATE_WCB
                                                  .EXTRN  SMG$_INVWIDARG, SMG$_INVPAGARG
                                                  .EXTRN  SMG$_INVCOLARG, SMG$_PBDIN_USE
                                                  .EXTRN  SMG$GET_TERM_DATA

                                                  .PSECT  _SMG$CODE,NOWRT,  SHR,  PIC,2

                               0FFC 00000         .ENTRY  SMG$CHANGE_PBD_CHARACTERISTICS, Save R2,R3,-; 0535
                                                          R4,R5,R6,R7,R8,R9,R10,R11                   ;
              5B 00000000G   00  9E 00002         MOVAB   SMG$$OUTPUT, R11                            ;
              5A 00000000G   00  9E 00009         MOVAB   SMG$GET_TERM_DATA, R10                      ;
              5E             14  C2 00010         SUBL2   #20, SP                                     ;
    50        6C             01  83 00013         SUBB3   #1, (AP), DIFF                              ; 0702
              06             50  91 00017         CMPB    DIFF, #6                                    ;
                             08  1B 0001A         BLEQU   1$                                          ;
              50 00000000G   8F  D0 0001C         MOVL    #SMG$_WRONUMARG, R0                         ;
                                 04 00023         RET                                                 ;
              50         04  BC  D0 00024  1$:    MOVL    @PBID, R0                                   ; 0704
                             11  19 00028         BLSS    2$                                          ;
       00000000'  EF         50  D1 0002A         CMPL    R0, PBD_L_COUNT                             ;
                             08  14 00031         BGTR    2$                                          ;
    08 00000000'  EF         50  E0 00033         BBS     R0, PBD_V_PB_AVAIL, 3$                      ;
              50 00000000G   8F  D0 0003B  2$:    MOVL    #SMG$_INVPAS_ID, R0                         ;
                                 04 00042         RET                                                 ;
              54 00000000'EF40  D0 00043  3$:     MOVL    PBD_A_PBCB[R0], PBCB                        ;
                             59  D4 0004B         CLRL    PASTING_PACKET_PANIC                        ; 0706
              02             6C  91 0004D         CMPB    (AP), #2                                    ; 0712
                             03  1E 00050         BGEQU   5$                                          ;
                           0189  31 00052  4$:    BRW     33$                                         ;
                         08  AC  D5 00055  5$:    TSTL    8(AP)                                       ;
                             F8  13 00058         BEQL    4$                                          ;
              58         08  BC  D0 0005A         MOVL    @P_DESIRED_WIDTH, R8                        ; 0719
                             08  12 0005E         BNEQ    6$                                          ;
              50 00000000G   8F  D0 00060         MOVL    #SMG$_INVWIDARG, R0                         ; 0720
                                 04 00067         RET                                                 ;
                       00A4  C4  D5 00068  6$:    TSTL    164(PBCB)                                   ; 0730
                             03  13 0006C         BEQL    7$                                          ;
                           018C  31 0006E         BRW     35$                                         ;
              53       0108  C4  9E 00071  7$:    MOVAB   264(PBCB), R3                               ; 0772
              52       00FC  C4  9E 00076         MOVAB   252(PBCB), R2                               ;
                             62  D5 0007B         TSTL    (R2)                                        ;
                             04  12 0007D         BNEQ    8$                                          ;
                             63  D4 0007F         CLRL    (R3)                                        ;
```

```
                               21  11 00081         BRB     9$                              ;
                       08      AE  D4 00083 8$:     CLRL    INPUT_ARGS                      ;
                       08      AE  9F 00086         PUSHAB  INPUT_ARGS                      ;
                     0104      C4  DD 00089         PUSHL   260(PBCB)                       ;
                               53  DD 0008D         PUSHL   R3                              ;
                     0100      C4  9F 0008F         PUSHAB  256(PBCB)                       ;
       14      AE    01DA      8F  3C 00093         MOVZWL  #474, 20(SP)                    ;
                       14      AE  9F 00099         PUSHAB  20(SP)                          ;
                               52  DD 0009C         PUSHL   R2                              ;
               6A              06  FB 0009E         CALLS   #6, SMG$GET_TERM_DATA           ;
               3A              50  E9 000A1         BLBC    STATUS, 12$                     ;
                               63  D5 000A4 9$:     TSTL    (R3)                            ; 0773
                               11  13 000A6         BEQL    10$                             ;
                     0104      C4  DD 000A8         PUSHL   260(PBCB)                       ; 0776
                               63  DD 000AC         PUSHL   (R3)                            ; 0775
                               54  DD 000AE         PUSHL   PBCB                            ;
               6B              03  FB 000B0         CALLS   #3, SMG$$OUTPUT                 ;
               55              50  D0 000B3         MOVL    R0, STATUS                      ;
               3F              55  E9 000B6         BLBC    STATUS, 14$                     ; 0777
                               62  D5 000B9 10$:    TSTL    (R2)                            ; 0784
                               04  12 000BB         BNEQ    11$                             ;
                               63  D4 000BD         CLRL    (R3)                            ;
                               20  11 000BF         BRB     13$                             ;
                       08      AE  D4 000C1 11$:    CLRL    INPUT_ARGS                      ;
                       08      AE  9F 000C4         PUSHAB  INPUT_ARGS                      ;
                     0104      C4  DD 000C7         PUSHL   260(PBCB)                       ;
                               53  DD 000CB         PUSHL   R3                              ;
                     0100      C4  9F 000CD         PUSHAB  256(PBCB)                       ;
       14      AE      DD      8F  9A 000D1         MOVZBL  #221, 20(SP)                    ;
                       14      AE  9F 000D6         PUSHAB  20(SP)                          ;
                               52  DD 000D9         PUSHL   R2                              ;
               6A              06  FB 000DB         CALLS   #6, SMG$GET_TERM_DATA           ;
               49              50  E9 000DE 12$:    BLBC    STATUS, 18$                     ;
                               63  D5 000E1 13$:    TSTL    (R3)                            ; 0785
                               1B  13 000E3         BEQL    15$                             ;
               56    0104      C4  D0 000E5         MOVL    260(PBCB), R6                   ; 0787
                     0104      C4  DD 000EA         PUSHL   260(PBCB)                       ; 0789
                               63  DD 000EE         PUSHL   (R3)                            ; 0788
                               54  DD 000F0         PUSHL   PBCB                            ;
               6B              03  FB 000F2         CALLS   #3, SMG$$OUTPUT                 ;
               55              50  D0 000F5         MOVL    R0, STATUS                      ;
               49              55  E9 000F8 14$:    BLBC    STATUS, 20$                     ; 0790
               57              66  D0 000FB         MOVL    (R6), NORMAL_WIDTH              ; 0791
                               04  11 000FE         BRB     16$                             ;
               57      50      8F  9A 00100 15$:    MOVZBL  #80, NORMAL_WIDTH               ; 0793
                               62  D5 00104 16$:    TSTL    (R2)                            ; 0799
                               04  12 00106         BNEQ    17$                             ;
                               63  D4 00108         CLRL    (R3)                            ;
                               21  11 0010A         BRB     19$                             ;
                       08      AE  D4 0010C 17$:    CLRL    INPUT_ARGS                      ;
                       08      AE  9F 0010F         PUSHAB  INPUT_ARGS                      ;
                     0104      C4  DD 00112         PUSHL   260(PBCB)                       ;
                               53  DD 00116         PUSHL   R3                              ;
                     0100      C4  9F 00118         PUSHAB  256(PBCB)                       ;
       14      AE    0246      8F  3C 0011C         MOVZWL  #582, 20(SP)                    ;
                       14      AE  9F 00122         PUSHAB  20(SP)                          ;
                               52  DD 00125         PUSHL   R2                              ;
```

```
                                                                    D 13
SMG$DISPLAY_LIN SMG$DISPLAY_LINKS - Virtual Display Linkages     16-Sep-1984 00:29:22    VAX-11 Bliss-32 V4.0-742          Page  20
1-096           SMG$CHANGE_PBD_CHARACTERISTICS                   14-Sep-1984 13:09:43    [SMGRTL.SRC]SMGDISLIN.B32;1        (6)

        6A              06  FB 00127          CALLS   #6, SMG$GET_TERM_DATA                  ;
        7C              50  E9 0012A  18$:    BLBC    STATUS, 27$                            ;
                        63  D5 0012D  19$:    TSTL    (R3)                                   ; 0800
                        1B  13 0012F          BEQL    21$                                    ;
        56      0104    C4  D0 00131          MOVL    260(PBCB), R6                          ; 0802
                0104    C4  DD 00136          PUSHL   260(PBCB)                              ; 0804
                        63  DD 0013A          PUSHL   (R3)                                   ; 0803
                        54  DD 0013C          PUSHL   PBCB                                   ;
        6B              03  FB 0013E          CALLS   #3, SMG$$OUTPUT                        ;
        55              50  D0 00141          MOVL    R0, STATUS                             ;
        7B              55  E9 00144  20$:    BLBC    STATUS, 30$                            ; 0805
        56              66  D0 00147          MOVL    (R6), WIDE_WIDTH                       ; 0806
                        04  11 0014A          BRB     22$                                    ;
        56        50    8F  9A 0014C  21$:    MOVZBL  #80, WIDE_WIDTH                        ; 0808
        57              58  D1 00150  22$:    CMPL    R8, NORMAL_WIDTH                       ; 0814
                        2E  14 00153          BGTR    25$                                    ;
                        62  D5 00155          TSTL    (R2)                                   ; 0816
                        04  12 00157          BNEQ    23$                                    ;
                        63  D4 00159          CLRL    (R3)                                   ;
                        21  11 0015B          BRB     24$                                    ;
                  08    AE  D4 0015D  23$:    CLRL    INPUT_ARGS                             ;
                  08    AE  9F 00160          PUSHAB  INPUT_ARGS                             ;
                0104    C4  DD 00163          PUSHL   260(PBCB)                              ;
                        53  DD 00167          PUSHL   R3                                     ;
                0100    C4  9F 00169          PUSHAB  256(PBCB)                              ;
  14    AE      0245    8F  3C 0016D          MOVZWL  #581, 20(SP)                           ;
                  14    AE  9F 00173          PUSHAB  20(SP)                                 ;
                        52  DD 00176          PUSHL   R2                                     ;
        6A              06  FB 00178          CALLS   #6, SMG$GET_TERM_DATA                  ;
        2B              50  E9 0017B          BLBC    STATUS, 27$                            ;
        52              57  D0 0017E  24$:    MOVL    NORMAL_WIDTH, RESULTANT_WIDTH          ; 0817
                        2D  11 00181          BRB     29$                                    ;
                        62  D5 00183  25$:    TSTL    (R2)                                   ; 0820
                        04  12 00185          BNEQ    26$                                    ;
                        63  D4 00187          CLRL    (R3)                                   ;
                        22  11 00189          BRB     28$                                    ;
                  08    AE  D4 0018B  26$:    CLRL    INPUT_ARGS                             ;
                  08    AE  9F 0018E          PUSHAB  INPUT_ARGS                             ;
                0104    C4  DD 00191          PUSHL   260(PBCB)                              ;
                        53  DD 00195          PUSHL   R3                                     ;
                0100    C4  9F 00197          PUSHAB  256(PBCB)                              ;
  14    AE      0246    8F  3C 0019B          MOVZWL  #582, 20(SP)                           ;
                  14    AE  9F 001A1          PUSHAB  20(SP)                                 ;
                        52  DD 001A4          PUSHL   R2                                     ;
        6A              06  FB 001A6          CALLS   #6, SMG$GET_TERM_DATA                  ;
        01              50  E8 001A9  27$:    BLBS    STATUS, 28$                            ;
                            04 001AC          RET                                            ;
        52              56  D0 001AD  28$:    MOVL    WIDE_WIDTH, RESULTANT_WIDTH            ; 0821
                        63  D5 001B0  29$:    TSTL    (R3)                                   ; 0824
                        11  13 001B2          BEQL    31$                                    ;
                0104    C4  DD 001B4          PUSHL   260(PBCB)                              ; 0827
                        63  DD 001B8          PUSHL   (R3)                                   ; 0826
                        54  DD 001BA          PUSHL   PBCB                                   ;
        6B              03  FB 001BC          CALLS   #3, SMG$$OUTPUT                        ;
        55              50  D0 001BF          MOVL    R0, STATUS                             ;
        72              55  E9 001C2  30$:    BLBC    STATUS, 39$                            ; 0828
        50              52  D0 001C5  31$:    MOVL    RESULTANT_WIDTH, R0                    ; 0837
```

```
                                                          E 13
SMG$DISPLAY_LIN SMG$DISPLAY_LINKS - Virtual Display Linkages      16-Sep-1984 00:29:22    VAX-11 Bliss-32 V4.0-742            Page  21
1-096           SMG$CHANGE_PBD_CHARACTERISTICS                    14-Sep-1984 13:09:43    [SMGRTL.SRC]SMGDISLIN.B32;1               (6)

                               58          50  D1 001C8       CMPL    R0, R8                                              ;
                                           03  1B 001CB       BLEQU   32$                                                 ;
                               50          58  D0 001CD       MOVL    R8, R0                                              ;
                               52          50  D0 001D0 32$:  MOVL    R0, RESULTANT_WIDTH                                 ;
                               50     5A   A4  3C 001D3       MOVZWL  90(PBCB), PREVIOUS_WIDTH                            ; 0853
                         5A    A4          52  B0 001D7       MOVW    RESULTANT_WIDTH, 90(PBCB)                           ; 0854
                               59          01  D0 001DB       MOVL    #1, PASTING_PACKET_PANIC                            ; 0863
                               03          6C  91 001DE 33$:  CMPB    (AP), #3                                            ; 0878
                                           0A  1F 001E1       BLSSU   34$                                                 ;
                                      0C   AC  D5 001E3       TSTL    12(AP)                                              ;
                                           05  13 001E6       BEQL    34$                                                 ;
                         0C    BC     5A   A4  3C 001E8       MOVZWL  90(PBCB), @P_RESULTING_WIDTH                        ; 0881
                               04          6C  91 001ED 34$:  CMPB    (AP), #4                                            ; 0892
                                           4F  1F 001F0       BLSSU   41$                                                 ;
                                      10   AC  D5 001F2       TSTL    16(AP)                                              ;
                                           4A  13 001F5       BEQL    41$                                                 ;
                                    00A4   C4  D5 001F7       TSTL    164(PBCB)                                           ; 0895
                                           08  13 001FB       BEQL    36$                                                 ;
                               50 00000000G 8F D0 001FD 35$:  MOVL    #SMG$_PBDIN_USE, R0                                 ; 0896
                                               04 00204       RET                                                         ;
                               52     10   BC  D0 00205 36$:  MOVL    @P_DESIRED_HEIGHT, R2                               ; 0897
                                           08  12 00209       BNEQ    37$                                                 ;
                               50 00000000G 8F D0 0020B       MOVL    #SMG$_INVPAGARG, R0                                 ; 0898
                                               04 00212       RET                                                         ;
      52      5F    A4         08          00  ED 00213 37$:  CMPZV   #0, #8, 95(PBCB), R2                                ; 0899
                                           26  13 00219       BEQL    41$                                                 ;
                               18          52  D1 0021B       CMPL    R2, #24                                             ; 0906
                                           03  1B 0021E       BLEQU   38$                                                 ;
                               52          18  D0 00220       MOVL    #24, R2                                             ;
      52      5F    A4         08          00  ED 00223 38$:  CMPZV   #0, #8, 95(PBCB), R2                                ;
                                           0F  1B 00229       BLEQU   40$                                                 ;
                                           54  DD 0022B       PUSHL   PBCB                                                ; 0908
                  00000000G    00          01  FB 0022D       CALLS   #1, SMG$$ERASE_PASTEBOARD                           ;
                               55          50  D0 00234       MOVL    R0, STATUS                                          ;
                               61          55  E9 00237 39$:  BLBC    STATUS, 44$                                         ; 0909
                         5F    A4          52  90 0023A 40$:  MOVB    R2, 95(PBCB)                                        ; 0914
                               59          01  D0 0023E       MOVL    #1, PASTING_PACKET_PANIC                            ; 0915
                               05          6C  91 00241 41$:  CMPB    (AP), #5                                            ; 0923
                                           0A  1F 00244       BLSSU   42$                                                 ;
                                      14   AC  D5 00246       TSTL    20(AP)                                              ;
                                           05  13 00249       BEQL    42$                                                 ;
                         14    BC     5F   A4  9A 0024B       MOVZBL  95(PBCB), @P_RESULTING_HEIGHT                       ; 0926
                               6C          59  E9 00250 42$:  BLBC    PASTING_PACKET_PANIC, 47$                           ; 0935
                                      08   A4  DD 00253       PUSHL   8(PBCB)                                             ; 0941
                  00000000G    00          01  FB 00256       CALLS   #1, SMG$$DEALLOCATE_WCB                             ;
                               55          50  D0 0025D       MOVL    R0, STATUS                                          ;
                               38          55  E9 00260       BLBC    STATUS, 44$                                         ; 0942
                                      08   A4  9F 00263       PUSHAB  8(PBCB)                                             ; 0948
                         08    AE     5A   A4  3C 00266       MOVZWL  90(PBCB), 8(SP)                                     ; 0947
                                      08   AE  9F 0026B       PUSHAB  8(SP)                                               ;
                         08    AE     5F   A4  9A 0026E       MOVZBL  95(PBCB), 8(SP)                                     ; 0946
                                      08   AE  9F 00273       PUSHAB  8(SP)                                               ;
                  00000000G    00          03  FB 00276       CALLS   #3, SMG$$CREATE_WCB                                 ; 0948
                               55          50  D0 0027D       MOVL    R0, STATUS                                          ;
                               18          55  E9 00280       BLBC    STATUS, 44$                                         ; 0949
                               53          64  D0 00283       MOVL    (PBCB), CURR_PP                                     ; 0955
                               54          53  D1 00286 43$:  CMPL    CURR_PP, PBCB                                       ; 0956
```

```
                                  1C  13 00289          BEQL    46$                                          ;
                    52        F8  A3  9E 0028B          MOVAB   -8(R3), PP_BASE                              ; 0961
                                  52  DD 0028F          PUSHL   PP_BASE                                      ; 0964
          00000000G 00            01  FB 00291          CALLS   #1, SMG$$CALC_PASTE_TRANSF                   ;
                    55            50  D0 00298          MOVL    R0, STATUS                                   ;
                    03            55  E8 0029B 44$:     BLBS    STATUS, 45$                                  ; 0965
                                0092  31 0029E          BRW     53$                                          ;
                    53        08  A2  D0 002A1 45$:     MOVL    8(PP_BASE), CURR_PP                          ; 0966
                                  DF  11 002A5          BRB     43$                                          ;
                                  54  DD 002A7 46$:     PUSHL   PBCB                                         ; 0973
          00000000G 00            01  FB 002A9          CALLS   #1, SMG$$CHECK_FOR_OUTPUT_PBCB               ;
                    55            50  D0 002B0          MOVL    R0, STATUS                                   ;
                    09            55  E8 002B3          BLBS    STATUS, 47$                                  ; 0974
                                  55  DD 002B6          PUSHL   STATUS                                       ;
          00000000G 00            01  FB 002B8          CALLS   #1, LIB$SIGNAL                               ;
                    06            6C  91 002BF 47$:     CMPB    (AP), #6                                     ; 0982
                                  79  1F 002C2          BLSSU   55$                                          ;
                              18  AC  D5 002C4          TSTL    24(AP)                                       ;
                                  74  13 002C7          BEQL    55$                                          ;
                    50      00FC  C4  9E 002C9          MOVAB   252(PBCB), R0                                ; 0988
                    52      0108  C4  9E 002CE          MOVAB   264(PBCB), R2                                ;
                    01        18  BC  D1 002D3          CMPL    @P_DESIRED_BACKGROUND_COLOR, #1              ; 0987
                                  1C  12 002D7          BNEQ    48$                                          ;
                                  60  D5 002D9          TSTL    (R0)                                         ; 0988
                                  1C  13 002DB          BEQL    49$                                          ;
                              08  AE  D4 002DD          CLRL    INPUT_ARGS                                   ;
                              08  AE  9F 002E0          PUSHAB  INPUT_ARGS                                   ;
                            0104  C4  DD 002E3          PUSHL   260(PBCB)                                    ;
                                  52  DD 002E7          PUSHL   R2                                           ;
                            0100  C4  9F 002E9          PUSHAB  256(PBCB)                                    ;
               14   AE      0228  8F  3C 002ED          MOVZWL  #552, 20(SP)                                 ;
                                  1E  11 002F3          BRB     51$                                          ;
                                  60  D5 002F5 48$:     TSTL    (R0)                                         ; 0989
                                  04  12 002F7          BNEQ    50$                                          ;
                                  62  D4 002F9 49$:     CLRL    (R2)                                         ;
                                  21  11 002FB          BRB     52$                                          ;
                              08  AE  D4 002FD 50$:     CLRL    INPUT_ARGS                                   ;
                              08  AE  9F 00300          PUSHAB  INPUT_ARGS                                   ;
                            0104  C4  DD 00303          PUSHL   260(PBCB)                                    ;
                                  52  DD 00307          PUSHL   R2                                           ;
                            0100  C4  9F 00309          PUSHAB  256(PBCB)                                    ;
               14   AE      01C8  8F  3C 0030D          MOVZWL  #456, 20(SP)                                 ;
                              14  AE  9F 00313 51$:     PUSHAB  20(SP)                                       ;
                                  50  DD 00316          PUSHL   R0                                           ;
                    6A            06  FB 00318          CALLS   #6, SMG$GET_TERM_DATA                        ;
                    32            50  E9 0031B          BLBC    STATUS, 57$                                  ;
                                  62  D5 0031E 52$:     TSTL    (R2)                                         ; 0991
                                  15  13 00320          BEQL    54$                                          ;
                            0104  C4  DD 00322          PUSHL   260(PBCB)                                    ; 0994
                                  62  DD 00326          PUSHL   (R2)                                         ; 0993
                                  54  DD 00328          PUSHL   PBCB                                         ;
                    6B            03  FB 0032A          CALLS   #3, SMG$$OUTPUT                              ;
                    55            50  D0 0032D          MOVL    R0, STATUS                                   ;
                    04            55  E8 00330          BLBS    STATUS, 54$                                  ; 0995
                    50            55  D0 00333 53$:     MOVL    STATUS, R0                                   ;
                                      04 00336          RET                                                  ;
               00F9 C4        18  BC  D0 00337 54$:     MOVL    @P_DESIRED_BACKGROUND_COLOR, 249(PBCB)       ; 0998
```

```
                              07            6C  91 0033D 55$:     CMPB    (AP), #7                                   ; 1006
                                            0B  1F 00340          BLSSU   56$                                        ;
                                      1C    AC  D5 00342          TSTL    28(AP)                                     ;
                                            06  13 00345          BEQL    56$                                        ;
                         1C   BC      00F9  C4  9A 00347          MOVZBL  249(PBCB), @P_RESULTING_BACKGROUND_COLOR   ; 1009
                              50            01  D0 0034D 56$:     MOVL    #1, R0                                     ; 1012
                                                04 00350 57$:     RET                                                ; 1014

; Routine Size:  849 bytes,    Routine Base:  _SMG$CODE + 0000
```

```
1 %SBTTL 'SMG$CHANGE_VIRTUAL_DISPLAY - Change Virtual Display'
1 GLOBAL ROUTINE SMG$CHANGE_VIRTUAL_DISPLAY (
1                                          DISPLAY_ID,
1                                          NUM_ROWS,
1                                          NUM_COLS,
1                                          DISPLAY_ATTRIBUTES,
1                                          VIDEO_ATTRIBUTES,
1                                          CHAR_SET
1                                         ) =
1 !++
1 ! FUNCTIONAL DESCRIPTIOM:
1 !
1 !      This routine changes the size or default attributes of an
1 !      existing virtual display.  The text which is currently in this
1 !      virtual display is remapped to fit the new dimensions starting
1 !      at row 1 column 1.  Resulting cursor position will be at row 1
1 !      column 1.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !      ret_status.wlc.v = SMG$CHANGE_VIRTUAL_DISPLAY (
1 !                               DISPLAY_ID.rl.r,
1 !                               [,NUM_ROWS.rl.r]
1 !                               [,NUM_COLS.rl.r]
1 !                               [,DISPLAY_ATTRIBUTES.rl.r]
1 !                               [,VIDEO_ATTRIBUTES.rl.r]
1 !                               [,CHAR_SET.rl.r])
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !      DISPLAY_ID.rl.r Display id of virtual display to be changed.
1 !
1 !      NUM_ROWS.rl.r   Number of rows in new virtual display.
1 !                      If omitted, the number of rows remains the same.
1 !
1 !      NUM_COLS.rl.r   Number of columns in new virtual display.
1 !                      If omitted, the number of columns remains the same.
1 !
1 !      DISPLAY_ATTRIBUTES.rl.r The default display attributes:
1 !
1 !                      SMG$M_BORDER if virtual display is to be
1 !                                   displayed with a border.
1 !
1 !                      SMG$M_TRUNC_ICON if an icon should be displayed
1 !                                   when text overflows the display bounds.
1 !
1 !                      SMG$M_DISPLAY_CONTROLS if carriage controls (CR, LF,
1 !                                   FF, VT, HT) should be displayed instead
1 !                                   of executed.
1 !
1 !                      If omitted, the default display attributes
1 !                      currently associated with the display will be
1 !                      retained.
1 !
1 !      VIDEO_ATTRIBUTES.rl.r   The default rendition code to be
1 !                      applied to all output to this display unless
1 !                      overridden on a particular output call.
```

```
1 !                             If omitted, the current video attributes are
1 !                             retained.
1 !
1 !                             Values:
1 !
1 !                             SMG$M_BLINK     displays characters blinking.
1 !
1 !                             SMG$M_BOLD      displays characters in
1 !                                             higher-than-normal intensity.
1 !
1 !                             SMG$M_REVERSE   displays characters in reverse
1 !                                             video -- that is, using the
1 !                                             opposite default rendition of
1 !                                             the virtual display.
1 !
1 !                             SMG$M_UNDERLINE displays characters underlined.
1 !
1 !             CHAR_SET.rl.r   The default character set for all text
1 !                             associated with this display.
1 !                             Recognized values are:
1 !                                             SMG$C_UNITED_KINGDOM
1 !                                             SMG$C_ASCII (default)
1 !                                             SMG$C_SPEC_GRAPHICS
1 !                                             SMG$C_ALT_CHAR
1 !                                             SMG$C_ALT_GRAPHICS
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !             NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !             NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !             SS$_NORMAL      Normal successful completion
1 !             LIB$_INSVIRMEM  Insufficient virtual memory to reallocate needed
1 !                             buffers.
1 !             SMG$_INVARG     Unrecognized Video Attributes
1 !                          or Unrecognized Display Attributes
1 !             SMG$_WRONUMARG  Wrong number of arguments.
1 !
1 ! SIDE EFFECTS:
1 !
1 !             Cursor for virtual display will be forced to row 1 column 1 if
1 !             display is redimensioned.
1 !             If a labeled border applies and does not fit newly redimensioned
1 !             display, the label will be deleted.
1 !--
2       BEGIN
2       BUILTIN
2           NULLPARAMETER;
2
2       LOCAL
2           STATUS,                 ! Status of subroutine calls
2           PP  : REF $PP_DECL,     ! Addr. of a pasting packet
```

```
                NEW_ROWS,                       ! New number of rows
                NEW_COLS,                       ! New number of columns
                DCB : REF $DCB_DECL,            ! Addr of display control block
                NEW_SIZE;                       ! New rows * columns

            $SMG$VALIDATE_ARGCOUNT (1, 6);      ! Test for right no. of args

            $SMG$GET_DCB (.DISPLAY_ID, DCB);    ! Get address of virtual display
                                                ! control block.

        !+
        ! Determine size of new buffer we need.
        !-
            IF NOT NULLPARAMETER (NUM_ROWS)     ! If new number of rows specified
            THEN
                NEW_ROWS = ..NUM_ROWS
            ELSE
                NEW_ROWS = .DCB [DCB_W_NO_ROWS];

            IF NOT NULLPARAMETER (NUM_COLS)     ! If new number of columns specified
            THEN
                NEW_COLS = ..NUM_COLS
            ELSE
                NEW_COLS = .DCB [DCB_W_NO_COLS];

            NEW_SIZE = .NEW_ROWS * .NEW_COLS;

        !+
        ! Adjust default display, video attributes and default character set if
        ! they are specified.
        !-
            IF NOT NULLPARAMETER (DISPLAY_ATTRIBUTES)   ! If display attributes specified
            THEN
                DCB [DCB_B_DEF_DISPLAY_ATTR] = ..DISPLAY_ATTRIBUTES;

            IF NOT NULLPARAMETER (VIDEO_ATTRIBUTES)     ! If video attributes specified
            THEN
                DCB [DCB_B_DEF_VIDEO_ATTR] = ..VIDEO_ATTRIBUTES;

            IF NOT NULLPARAMETER (CHAR_SET)     ! If char set specified
            THEN
                DCB [DCB_B_DEF_CHAR_SET] = ..CHAR_SET;

        !+
        ! If the dimensions of the old buffer and the new buffers are different,
        ! we will have to allocate new buffer space and copy existing text into
        ! new buffers.
        !-
            IF .DCB [DCB_L_BUFSIZE] NEQ .NEW_SIZE       OR
                .DCB [DCB_W_NO_ROWS] NEQ .NEW_ROWS      OR
                .DCB [DCB_W_NO_COLS] NEQ .NEW_COLS
            THEN
                BEGIN   ! Redimensioning required
                LOCAL
                    STATUS,                     ! Status of subroutine calls
                    ROWS_TO_MOVE,       ! No of rows that will be moved from
                                        ! old buffer to new.
```

```
3          COLS_TO_MOVE,        ! No of columns that will be moved from
3                               ! old to new.
3          NEW_TEXT_BUF : REF VECTOR [,BYTE],  ! Addr of new text
3                                              ! buffer
3          NEW_ATTR_BUF : REF VECTOR [,BYTE],  ! Addr of new attr
3                                              ! buffer
3          NEW_CHAR_BUF : REF VECTOR [,BYTE],  ! Addr of new char_set
3                                              ! buffer
3
3          TEXT_PTR : REF VECTOR [, BYTE],     ! Address of current
3                                              ! text buffer in DCB.
3
3          ATTR_PTR : REF VECTOR [,BYTE],      ! Address of current
3                                              ! attr buffer in DCB
3
3          CHAR_PTR : REF VECTOR [,BYTE];      ! Address of current
3                                              ! char_set buffer in
3                                              ! DCB
3
3      !+
3      ! Get space for two new, properly-dimensioned buffers.
3      !-
4      IF NOT (STATUS = LIB$GET_VM (%REF (2 * .NEW_SIZE),
4                                   NEW_TEXT_BUF))
3      THEN
3          RETURN (.STATUS);
3
3      NEW_ATTR_BUF = .NEW_TEXT_BUF + .NEW_SIZE;
3
3      !+
3      ! Now need to copy text and attribute information from
3      ! .DCB [DCB_A_TEXT_BUF] and .DCB [DCB_A_ATTR_BUF] to
3      ! .NEW_TEST_BUF and .NEW_ATTR_BUF, preserving the line context.
3      !       First pre-blank new text_buffer and attribute buffer in
3      !  case old do not cover new area.
3      !-
3      CH$FILL ( %C' ',                       .NEW_SIZE, .NEW_TEXT_BUF);
3      CH$FILL ( .DCB [DCB_B_DEF_VIDEO_ATTR], .NEW_SIZE, .NEW_ATTR_BUF);
3
3      TEXT_PTR = .DCB [DCB_A_TEXT_BUF];
3      ATTR_PTR = .DCB [DCB_A_ATTR_BUF];
3      CHAR_PTR = .DCB [DCB_A_CHAR_SET_BUF];
3
3      ROWS_TO_MOVE = MIN (.DCB [DCB_W_NO_ROWS], .NEW_ROWS);
3      COLS_TO_MOVE = MIN (.DCB [DCB_W_NO_COLS], .NEW_COLS);
3
3      INCR I FROM 1 TO .ROWS_TO_MOVE
3      DO
4          BEGIN       ! Move text and attrib. to new buffers.
4          LOCAL
4              SOURCE_INDEX,
4              DEST_INDEX;
4
4          SOURCE_INDEX = (.I -1) * .DCB [DCB_W_NO_COLS] ;
4          DEST_INDEX   = (.I -1) * ..NUM_COLS          ;
4
4          CH$MOVE ( .COLS_TO_MOVE,                    ! No of chars.
4                           TEXT_PTR [.SOURCE_INDEX],          ! From
4                           NEW_TEXT_BUF [.DEST_INDEX]);       ! To
4
4                   CH$MOVE ( .COLS_TO_MOVE,                   ! No. of chars.
4                           ATTR_PTR [.SOURCE_INDEX],          ! From
4                           NEW_ATTR_BUF [.DEST_INDEX]);       ! To
4
4
3                   END;        ! Move text and attrib to new buffers.
3
3               !+
3               ! Deal with alternate character set buffers if they exist.
3               !-
3               IF .DCB [DCB_A_CHAR_SET_BUF] NEQ 0
3               THEN
4                   BEGIN       ! Alt. char set buffer exists
4                   !+
4                   ! Allocate a new alternate character set buffer and init. it
4                   !-
5                   IF NOT (STATUS = LIB$GET_VM (NEW_SIZE, NEW_CHAR_BUF))
4                   THEN
5                       BEGIN
5                       LIB$FREE_VM (%REF (2* .NEW_SIZE), NEW_TEXT_BUF);
5                       RETURN (.STATUS); ! Return LIB$_INSVIRMEM from GET call
4                       END;
4
4                   CH$FILL ( .DCB [DCB_B_DEF_CHAR_SET], .NEW_SIZE,
4                           .NEW_CHAR_BUF);
4
4                   !+
4                   ! Move current contents row by row
4                   !-
4                   INCR I FROM 1 TO .ROWS_TO_MOVE
4                   DO
5                       BEGIN   ! Move loop
5                       LOCAL
5                           SOURCE_INDEX,
5                           DEST_INDEX;
5
5                       SOURCE_INDEX = (.I-1) * .DCB [DCB_W_NO_COLS] ;
5                       DEST_INDEX   = (.I-1) * ..NUM_COLS ;
5                       CH$MOVE ( .COLS_TO_MOVE,               ! No of chars
5                               CHAR_PTR [.SOURCE_INDEX],      ! From
5                               NEW_CHAR_BUF [.DEST_INDEX]);   ! To
4                       END;    ! Move loop
4
4                   !+
4                   ! Free old alternate char. set buffer and plug in new addr.
4                   !-
5                   IF NOT (STATUS = LIB$FREE_VM ( DCB [DCB_L_BUFSIZE],
5                                           DCB [DCB_A_CHAR_SET_BUF]))
4                   THEN
4                       RETURN (.STATUS);
4
4                   DCB [DCB_A_CHAR_SET_BUF] = .NEW_CHAR_BUF;
4
3                   END;        ! Alt. char set buffer exists
```

```
3
3         !+
3         ! Now that the text and attributes are safe in their
3         ! new buffers, we release the old buffers and put the addresses
3         ! of the new buffers in the DCB.
3         !-
4         IF NOT (STATUS = LIB$FREE_VM (%REF ( 2 * .DCB [DCB_L_BUFSIZE]),
4                                       DCB [DCB_A_TEXT_BUF]))
3         THEN
3             RETURN (.STATUS);
3
3         DCB [DCB_A_TEXT_BUF] = .NEW_TEXT_BUF;   ! Plug in new addresses
3         DCB [DCB_A_ATTR_BUF] = .NEW_ATTR_BUF;
3
3         !+
3         ! If the number of rows changed, we need to reallocate the
3         ! line characteristics vector and copy over as much of it as
3         ! fits.
3         !-
3         IF .DCB [DCB_W_NO_ROWS] NEQ .NEW_ROWS
3         THEN
4             BEGIN       ! No. of rows changed
4             LOCAL
4                 NEW_LINE_CHAR : REF VECTOR [,BYTE],      ! Addr of a new
4                                                          ! line Char.
4                                                          ! vector
4
4                 LINE_CHAR_PTR : REF VECTOR [,BYTE];      ! Addr of curr.
4                                                          ! line Char.
4                                                          ! vector
4
4             LINE_CHAR_PTR = .DCB [DCB_A_LINE_CHAR];
4
4             !+
4             ! Allocate a new line characteristics vector of the right
4             ! length.  Quit if we can't get it.
4             !-
5             IF NOT (STATUS = LIB$GET_VM ( %REF (.NEW_ROWS + 1),
5                                           NEW_LINE_CHAR))
4             THEN
5                 BEGIN   ! Error path
5                 !+
5                 ! Give back all space acquired on this transaction,
5                 ! ignoring further errors, and quit.
5                 !-
5                 LIB$FREE_VM ( %REF (2 * .NEW_SIZE), NEW_TEXT_BUF);
5                 RETURN ( .STATUS );
4                 END;    ! Error path
4
4             !+
4             ! Clear entire allocated vector to zero
4             !-
4             CH$FILL ( 0, .NEW_ROWS+1, .NEW_LINE_CHAR);
4
4             !+
4             ! Copy over as much of old line characteristics vector as
4             ! will fit.
```

```
4            !-
4            CH$MOVE ( .ROWS_TO_MOVE,
4                      LINE_CHAR_PTR [1],
4                      NEW_LINE_CHAR [1]);
4
4            !+
4            ! Free former line characteristics vector.
4            !-
5            IF NOT (STATUS = LIB$FREE_VM (
5                                %REF (.DCB [DCB_W_NO_ROWS] +1),
5                                DCB [DCB_A_LINE_CHAR]))
4            THEN
4                RETURN (.STATUS);
4
4            !+
4            ! Store address of new line characteristics vector in DCB
4            !-
4            DCB [DCB_A_LINE_CHAR] = .NEW_LINE_CHAR;
3            END;         ! No. of rows changed
3
3        !+
3        ! Adjust the no. of rows and no. of cols. recorded in the DCB.
3        !-
3        DCB [DCB_W_NO_ROWS] = .NEW_ROWS;         ! Adjust row/column size
3        DCB [DCB_W_NO_COLS] = .NEW_COLS;
3        DCB [DCB_L_BUFSIZE] = .NEW_SIZE;
3
3        !+
3        ! Force cursor to home.
3        !-
3        DCB [DCB_W_CURSOR_ROW] = 1;
3        DCB [DCB_W_CURSOR_COL] = 1;
3
3        !+
3        ! Knock down flags that indicate we are at end of a row and that
3        ! we are in last line.
3        !-
3        DCB [DCB_V_FULL]   = 0;
3        DCB [DCB_V_COL_80] = 0;
3
3        !+
3        ! Reset the scrolling region within the redimensioned virtual
3        ! display to be the whole display.
3        !-
3        DCB [DCB_W_TOP_OF_SCRREG]    = 1;
3        DCB [DCB_W_BOTTOM_OF_SCRREG] = .NEW_ROWS;
3
3        !+
3        ! Now deal with border data, if any exists.
3        !-
3        IF .DCB [DCB_V_BORDERED]
3        THEN
4            BEGIN         ! Bordered
4            LOCAL
4                DESC : REF BLOCK [8,BYTE]; ! Pointer to dynamic string
4                                           ! desc. for border label
4
```

```
          DESC = DCB [DCB_Q_LABEL_DESC];
          IF .DESC [DSC$A_POINTER] NEQ 0       ! If label exists
          THEN
              BEGIN   ! Label exists
              LOCAL
                  TEMP;
              TEMP = .DCB [DCB_W_LABEL_UNITS];

              !+
              ! Try to reapply our existing border label on this
              ! redimensioned virtual display.  If it now doesn't
              ! fit because of the new dimensions, delete the label.
              !-
              IF NOT (SMG$LABEL_BORDER (
                              .DISPLAY_ID,
                              .DESC,
                              %REF (.DCB [DCB_B_LABEL_POS]),
                              !+
                              ! Conditionalize UNITS parameter to
                              ! LABEL_BORDER depending on whether
                              ! caller originally specified
                              ! "centering" or gave us specific units.
                              !-
                              (IF .DCB [DCB_V_LABEL_CENTER] THEN 0
                                                            ELSE TEMP),
                              %REF ( .DCB [DCB_B_LABEL_REND])
                              ))
              THEN
                  LIB$SFREE1_DD ( .DESC);      ! Delete label
              END;      ! Label exists
          END;          ! Bordered
      END;    ! Redimensioning required

!+
! Since the dimension of the virtual display may have changed, or we
! may have added or deleted a border,  we need to recalculate the
! transformation constants that occur in each pasting packet we are
! involved in.
! Check to see if we can do it now or must wait because we are batched.
!-
  IF .DCB [DCB_L_BATCH_LEVEL] EQL 0
  THEN
      BEGIN   ! Can do it now
      LOCAL
          CURR_PP : REF $PP_DECL;      ! Addr of a pasting packet

      IF NOT (STATUS = SMG$$RECALC_PP_FIELDS ( .DCB))
      THEN
          RETURN (.STATUS) ;

      !+
      ! Remap all pasteboard buffers to which we are pasted, from the
      ! bottom outward.
      !-
      CURR_PP = .DCB [DCB_A_PP_NEXT];
      WHILE .CURR_PP NEQ DCB [DCB_A_PP_NEXT]
      DO
```

```
; 1216          1471  4             BEGIN       ! Remap all pasteboards
; 1217          1472  4             LOCAL
; 1218          1473  4                 PBCB : REF $PBCB_DECL;  ! Addr of a pasteboard control
; 1219          1474  4                                         ! block
; 1220          1475  4             PBCB = .CURR_PP [PP_A_PBCB_ADDR];
; 1221          1476  5             IF NOT (STATUS = SMG$$CHECK_FOR_OUTPUT_PBCB ( .PBCB))
; 1222          1477  4             THEN
; 1223          1478  4                 RETURN ( .STATUS);      ! Quit if any one of them fails
; 1224          1479  4
; 1225          1480  4             CURR_PP = .CURR_PP [PP_A_NEXT_DCB]; ! To next pasting packet
; 1226          1481  3             END;        ! Remap all pasteboards
; 1227          1482  3
; 1228          1483  3         RETURN ( SS$_NORMAL);
; 1229          1484  3         END     ! Can do it now
; 1230          1485  3
; 1231          1486  2     ELSE
; 1232          1487  2
; 1233          1488  3         BEGIN   ! Must delay until end_display_batch
; 1234          1489  3         DCB [DCB_V_PP_MISMATCH] = 1;    ! Mark it for later update
; 1235          1490  2         END;    ! Must delay until end_display_batch
; 1236          1491  2
; 1237          1492  2     RETURN ( SS$_NORMAL);
; 1238          1493  1     END;                        ! Routine SMG$CHANGE_VIRTUAL_DISPLAY



                                                  0FFC 00000          .ENTRY  SMG$CHANGE_VIRTUAL_DISPLAY, Save R2,R3,R4,- ; 1016
                                                                              R5,R6,R7,R8,R9,R10,R11                      ;
                              5E             30   C2 00002            SUBL2   #48, SP                                     ;
          50                  6C             01   83 00005            SUBB3   #1, (AP), DIFF                              ; 1134
                              05             50   91 00009            CMPB    DIFF, #5                                    ;
                                             08   1B 0000C            BLEQU   1$                                          ;
                              50 00000000G   8F   D0 0000E            MOVL    #SMG$_WRONUMARG, R0                         ;
                                             04   00015               RET                                                 ;
                              50        04   BC   D0 00016 1$:        MOVL    @DISPLAY_ID, R0                             ; 1136
                         04   BC        38   A0   D1 0001A            CMPL    56(R0), @DISPLAY_ID                         ;
                                             06   12 0001F            BNEQ    2$                                          ;
                              11        44   A0   91 00021            CMPB    68(R0), #17                                 ;
                                             08   13 00025            BEQL    3$                                          ;
                              50 00000000G   8F   D0 00027 2$:        MOVL    #SMG$_INVDIS_ID, R0                         ;
                                             04   0002E               RET                                                 ;
                              56        04   BC   D0 0002F 3$:        MOVL    @DISPLAY_ID, DCB                            ;
                              02             6C   91 00033            CMPB    (AP), #2                                    ; 1142
                                             0B   1F 00036            BLSSU   4$                                          ;
                                        08   AC   D5 00038            TSTL    8(AP)                                       ;
                                             06   13 0003B            BEQL    4$                                          ;
                              58        08   BC   D0 0003D            MOVL    @NUM_ROWS, NEW_ROWS                         ; 1144
                                             04   11 00041            BRB     5$                                          ;
                              58        02   A6   3C 00043 4$:        MOVZWL  2(DCB), NEW_ROWS                            ; 1146
                              03             6C   91 00047 5$:        CMPB    (AP), #3                                    ; 1148
                                             0C   1F 0004A            BLSSU   6$                                          ;
                                        0C   AC   D5 0004C            TSTL    12(AP)                                      ;
                                             07   13 0004F            BEQL    6$                                          ;
                         18   AE        0C   BC   D0 00051            MOVL    @NUM_COLS, NEW_COLS                         ; 1150
                                             05   11 00056            BRB     7$                                          ;
```

```
                            18  AE      06  A6  3C 00058 6$:     MOVZWL  6(DCB), NEW_COLS                                 ; 1152
              20  AE            58      18  AE  C5 0005D 7$:     MULL3   NEW_COLS, NEW_ROWS, NEW_SIZE                     ; 1154
                                04          6C  91 00063         CMPB    (AP), #4                                         ; 1160
                                            0A  1F 00066         BLSSU   8$                                               ;
                                        10  AC  D5 00068         TSTL    16(AP)                                           ;
                                            05  13 0006B         BEQL    8$                                               ;
                            2F  A6      10  BC  90 0006D         MOVB    @DISPLAY_ATTRIBUTES, 47(DCB)                     ; 1162
                                05          6C  91 00072 8$:     CMPB    (AP), #5                                         ; 1164
                                            0A  1F 00075         BLSSU   9$                                               ;
                                        14  AC  D5 00077         TSTL    20(AP)                                           ;
                                            05  13 0007A         BEQL    9$                                               ;
                            2E  A6      14  BC  90 0007C         MOVB    @VIDEO_ATTRIBUTES, 46(DCB)                       ; 1166
                                06          6C  91 00081 9$:     CMPB    (AP), #6                                         ; 1168
                                            0A  1F 00084         BLSSU   10$                                              ;
                                        18  AC  D5 00086         TSTL    24(AP)                                           ;
                                            05  13 00089         BEQL    10$                                              ;
                            30  A6      18  BC  90 0008B         MOVB    @CHAR_SET, 48(DCB)                               ; 1170
                                57      20  AE  D0 00090 10$:    MOVL    NEW_SIZE, R7                                     ; 1177
                                57      3C  A6  D1 00094         CMPL    60(DCB), R7                                      ;
                                            14  12 00098         BNEQ    11$                                              ;
        58    02  A6            10          00  ED 0009A         CMPZV   #0, #16, 2(DCB), NEW_ROWS                        ; 1178
                                            0C  12 000A0         BNEQ    11$                                              ;
   18   AE    06  A6            10          00  ED 000A2         CMPZV   #0, #16, 6(DCB), NEW_COLS                        ; 1179
                                            03  12 000A9         BNEQ    11$                                              ;
                                          01F7  31 000AB         BRW     29$                                              ;
                                        28  AE  9F 000AE 11$:    PUSHAB  NEW_TEXT_BUF                                     ; 1208
              18  AE            57          01  78 000B1         ASHL    #1, R7, 24(SP)                                   ;
                                        18  AE  9F 000B6         PUSHAB  24(SP)                                           ;
                      00000000G 00          02  FB 000B9         CALLS   #2, LIB$GET_VM                                   ;
                                6E          50  D0 000C0         MOVL    R0, STATUS                                       ;
                                03          6E  E8 000C3         BLBS    STATUS, 12$                                      ;
                                          016B  31 000C6         BRW     24$                                              ;
                            0C  AE     28 BE47  9E 000C9 12$:    MOVAB   @NEW_TEXT_BUF[R7], NEW_ATTR_BUF                  ; 1213
        57        20            6E          00  2C 000CF         MOVC5   #0, (SP), #32, R7, @NEW_TEXT_BUF                 ; 1222
                                        28  BE     000D4                                                                  ;
        57    2E  A6            6E          00  2C 000D6         MOVC5   #0, (SP), 46(DCB), R7, @NEW_ATTR_BUF             ; 1223
                                        0C  BE     000DC                                                                  ;
                                5B      10  A6  D0 000DE         MOVL    16(DCB), TEXT_PTR                                ; 1225
                                5A      14  A6  D0 000E2         MOVL    20(DCB), ATTR_PTR                                ; 1226
                            04  AE      18  A6  D0 000E6         MOVL    24(DCB), CHAR_PTR                                ; 1227
                                52      02  A6  3C 000EB         MOVZWL  2(DCB), R2                                       ; 1229
                                58          52  D1 000EF         CMPL    R2, NEW_ROWS                                     ;
                                            03  15 000F2         BLEQ    13$                                              ;
                                52          58  D0 000F4         MOVL    NEW_ROWS, R2                                     ;
                            14  AE          52  D0 000F7 13$:    MOVL    R2, ROWS_TO_MOVE                                 ;
                                52      06  A6  3C 000FB         MOVZWL  6(DCB), R2                                       ; 1230
                            18  AE          52  D1 000FF         CMPL    R2, NEW_COLS                                     ;
                                            04  15 00103         BLEQ    14$                                              ;
                                52      18  AE  D0 00105         MOVL    NEW_COLS, R2                                     ;
                            10  AE          52  D0 00109 14$:    MOVL    R2, COLS_TO_MOVE                                 ;
                                            57  D4 0010D         CLRL    I                                                ; 1247
                                            26  11 0010F         BRB     16$                                              ;
                                50      FF  A7  9E 00111 15$:    MOVAB   -1(R7), R0                                       ; 1239
                                51      06  A6  3C 00115         MOVZWL  6(DCB), R1                                       ;
              08  AE            50          51  C5 00119         MULL3   R1, R0, SOURCE_INDEX                             ;
                  59            50      0C  BC  C5 0011E         MULL3   @NUM_COLS, R0, DEST_INDEX                        ; 1240
           28 BE49         08 BE4B      10  AE  28 00123         MOVC3   COLS_TO_MOVE, @SOURCE_INDEX[TEXT_PTR], -         ; 1244
```

```
                                                                  E 14
SMG$DISPLAY_LIN SMG$DISPLAY_LINKS - Virtual Display Linkages    16-Sep-1984 00:29:22    VAX-11 Bliss-32 V4.0-742                    Page  34
1-096           SMG$CHANGE_VIRTUAL_DISPLAY - Change Virtual Dis 14-Sep-1984 13:09:43    [SMGRTL.SRC]SMGDISLIN.B32;1                      (7)

                                                                                   @NEW_TEXT_BUF[DEST_INDEX]                    ;
                                                0C  BE49  9F 0012C         PUSHAB  @NEW_ATTR_BUF[R9]                            ; 1248
                         9E       0C BE4A       14    AE  28 00130         MOVC3   COLS_TO_MOVE, @SOURCE_INDEX[ATTR_PTR], -     ;
                                                                                   @(SP)+                                       ;
                         D5            57       14    AE  F3 00137 16$:    AOBLEQ  ROWS_TO_MOVE, I, 15$                         ; 1232
                                                18    A6  D5 0013C         TSTL    24(DCB)                                      ; 1256
                                                      6E  13 0013F         BEQL    22$                                          ;
                                                1C    AE  9F 00141         PUSHAB  NEW_CHAR_BUF                                 ; 1262
                                                24    AE  9F 00144         PUSHAB  NEW_SIZE                                     ;
                            00000000G  00             02  FB 00147         CALLS   #2, LIB$GET_VM                               ;
                                       6E             50  D0 0014E         MOVL    R0, STATUS                                   ;
                                       16             6E  E8 00151         BLBS    STATUS, 19$                                  ;
                                                28    AE  9F 00154         PUSHAB  NEW_TEXT_BUF                                 ; 1265
                    0C   AE       24   AE             01  78 00157         ASHL    #1, NEW_SIZE, 12(SP)                         ;
                                                0C    AE  9F 0015D         PUSHAB  12(SP)                                       ;
                            00000000G  00             02  FB 00160 17$:    CALLS   #2, LIB$FREE_VM                              ;
                                                    00CA  31 00167 18$:    BRW     24$                                          ; 1266
      20   AE       30   A6            6E             00  2C 0016A 19$:    MOVC5   #0, (SP), 48(DCB), NEW_SIZE, @NEW_CHAR_BUF   ; 1270
                                                1C    BE     00171                                                              ;
                                                      59  D4 00173         CLRL    I                                            ; 1285
                                                      1B  11 00175         BRB     21$                                          ;
                                       50       FF    A9  9E 00177 20$:    MOVAB   -1(R9), R0                                   ; 1282
                                       51       06    A6  3C 0017B         MOVZWL  6(DCB), SOURCE_INDEX                         ;
                                       51             50  C4 0017F         MULL2   R0, SOURCE_INDEX                             ;
                                       50       0C    BC  C4 00182         MULL2   @NUM_COLS, DEST_INDEX                        ; 1283
                                       57       04    AE  D0 00186         MOVL    CHAR_PTR, R7                                 ; 1286
                    1C BE40          6147       10    AE  28 0018A         MOVC3   COLS_TO_MOVE, (SOURCE_INDEX)[R7], -          ;
                                                                                   @NEW_CHAR_BUF[DEST_INDEX]                    ;
                         E0            59       14    AE  F3 00192 21$:    AOBLEQ  ROWS_TO_MOVE, I, 20$                         ; 1275
                                                18    A6  9F 00197         PUSHAB  24(DCB)                                      ; 1293
                                                3C    A6  9F 0019A         PUSHAB  60(DCB)                                      ; 1292
                            00000000G  00             02  FB 0019D         CALLS   #2, LIB$FREE_VM                              ; 1293
                                       6E             50  D0 001A4         MOVL    R0, STATUS                                   ;
                                       BD             6E  E9 001A7         BLBC    STATUS, 18$                                  ;
                                  18   A6       1C    AE  D0 001AA         MOVL    NEW_CHAR_BUF, 24(DCB)                        ; 1297
                                                10    A6  9F 001AF 22$:    PUSHAB  16(DCB)                                      ; 1307
                    14   AE       3C   A6             01  78 001B2         ASHL    #1, 60(DCB), 20(SP)                          ; 1306
                                                14    AE  9F 001B8         PUSHAB  20(SP)                                       ;
                            00000000G  00             02  FB 001BB         CALLS   #2, LIB$FREE_VM                              ; 1307
                                       6E             50  D0 001C2         MOVL    R0, STATUS                                   ;
                                       6C             6E  E9 001C5         BLBC    STATUS, 24$                                  ;
                                  10   A6       28    AE  D0 001C8         MOVL    NEW_TEXT_BUF, 16(DCB)                        ; 1311
                                  14   A6       0C    AE  D0 001CD         MOVL    NEW_ATTR_BUF, 20(DCB)                        ; 1312
           58       02   A6            10             00  ED 001D2         CMPZV   #0, #16, 2(DCB), NEW_ROWS                    ; 1319
                                                      62  13 001D8         BEQL    26$                                          ;
                                       59       4C    A6  D0 001DA         MOVL    76(DCB), LINE_CHAR_PTR                       ; 1331
                                                24    AE  9F 001DE         PUSHAB  NEW_LINE_CHAR                                ; 1337
                                       52       01    A8  9E 001E1         MOVAB   1(R8), R2                                    ;
                                  14   AE             52  D0 001E5         MOVL    R2, 20(SP)                                   ;
                                                14    AE  9F 001E9         PUSHAB  20(SP)                                       ;
                            00000000G  00             02  FB 001EC         CALLS   #2, LIB$GET_VM                               ;
                                       6E             50  D0 001F3         MOVL    R0, STATUS                                   ;
                                       0F             6E  E8 001F6         BLBS    STATUS, 23$                                  ;
                                                28    AE  9F 001F9         PUSHAB  NEW_TEXT_BUF                                 ; 1345
                    14   AE       24   AE             01  78 001FC         ASHL    #1, NEW_SIZE, 20(SP)                         ;
                                                14    AE  9F 00202         PUSHAB  20(SP)                                       ;
                                                    FF58  31 00205         BRW     17$                                          ;
                            5A      24   AE  D0 00208 23$:   MOVL    NEW_LINE_CHAR, R10                              ; 1352
     52           00        6E           00  2C 0020C        MOVC5   #0, (SP), #0, R2, (R10)                         ;
                                         6A     00211                                                                ;
            01    AA     01 A9      14   AE  28 00212        MOVC3   ROWS_TO_MOVE, 1(LINE_CHAR_PTR), 1(R10)          ; 1360
                                    4C   A6  9F 00219        PUSHAB  76(DCB)                                         ; 1367
                      18    AE      02   A6  3C 0021C        MOVZWL  2(DCB), 24(SP)                                  ; 1366
                                    18   AE  D6 00221        INCL    24(SP)                                          ;
                                    18   AE  9F 00224        PUSHAB  24(SP)                                          ;
                   00000000G 00          02  FB 00227        CALLS   #2, LIB$FREE_VM                                 ; 1367
                            6E           50  D0 0022E        MOVL    R0, STATUS                                      ;
                            04           6E  E8 00231        BLBS    STATUS, 25$                                     ;
                            50           6E  D0 00234 24$:   MOVL    STATUS, R0                                      ; 1369
                                             04 00237        RET                                                     ;
                      4C    A6           5A  D0 00238 25$:   MOVL    R10, 76(DCB)                                    ; 1374
                      02    A6           58  B0 0023C 26$:   MOVW    NEW_ROWS, 2(DCB)                                ; 1380
                      06    A6      18   AE  B0 00240        MOVW    NEW_COLS, 6(DCB)                                ; 1381
                      3C    A6      20   AE  D0 00245        MOVL    NEW_SIZE, 60(DCB)                               ; 1382
                      28    A6 00010001  8F  D0 0024A        MOVL    #65537, 40(DCB)                                 ; 1387
                      34    A6           03  8A 00252        BICB2   #3, 52(DCB)                                     ; 1395
                      48    A6           01  B0 00256        MOVW    #1, 72(DCB)                                     ; 1401
                      4A    A6           58  B0 0025A        MOVW    NEW_ROWS, 74(DCB)                               ; 1402
                            43      2F   A6  E9 0025E        BLBC    47(DCB), 29$                                    ; 1407
                            52      08   A6  9E 00262        MOVAB   8(R6), DESC                                     ; 1414
                                    04   A2  D5 00266        TSTL    4(DESC)                                         ; 1415
                                         3A  13 00269        BEQL    29$                                             ;
                      2C    AE      2C   A6  3C 0026B        MOVZWL  44(DCB), TEMP                                   ; 1420
                      18    AE      33   A6  9A 00270        MOVZBL  51(DCB), 24(SP)                                 ; 1439
                                    18   AE  9F 00275        PUSHAB  24(SP)                                          ;
                  04  34    A6           02  E1 00278        BBC     #2, 52(DCB), 27$                                ; 1437
                                         7E  D4 0027D        CLRL    -(SP)                                           ;
                                         06  11 0027F        BRB     28$                                             ;
                            50      30   AE  9E 00281 27$:   MOVAB   TEMP, R0                                        ;
                                         50  DD 00285        PUSHL   R0                                              ;
                      1C    AE      31   A6  9A 00287 28$:   MOVZBL  49(DCB), 28(SP)                                 ; 1430
                                    1C   AE  9F 0028C        PUSHAB  28(SP)                                          ;
                                         52  DD 0028F        PUSHL   DESC                                            ; 1429
                                    04   AC  DD 00291        PUSHL   DISPLAY_ID                                      ; 1428
                       0000V CF          05  FB 00294        CALLS   #5, SMG$LABEL_BORDER                            ;
                             09          50  E8 00299        BLBS    R0, 29$                                         ;
                                         52  DD 0029C        PUSHL   DESC                                            ; 1442
                   00000000G 00          01  FB 0029E        CALLS   #1, LIB$SFREE1_DD                               ;
                                    1C   A6  D5 002A5 29$:   TSTL    28(DCB)                                         ; 1454
                                         2C  12 002A8        BNEQ    31$                                             ;
                                         56  DD 002AA        PUSHL   DCB                                             ; 1460
                       0000V CF          01  FB 002AC        CALLS   #1, SMG$$RECALC_PP_FIELDS                       ;
                             29          50  E9 002B1        BLBC    STATUS, 33$                                     ;
                             52     20   A6  D0 002B4        MOVL    32(DCB), CURR_PP                                ; 1468
                             51     20   A6  9E 002B8 30$:   MOVAB   32(DCB), R1                                     ; 1469
                             51          52  D1 002BC        CMPL    CURR_PP, R1                                     ;
                                         19  13 002BF        BEQL    32$                                             ;
                             51     14   A2  D0 002C1        MOVL    20(CURR_PP), PBCB                               ; 1475
                                         51  DD 002C5        PUSHL   PBCB                                            ; 1476
                   00000000G 00          01  FB 002C7        CALLS   #1, SMG$$CHECK_FOR_OUTPUT_PBCB                  ;
                             0C          50  E9 002CE        BLBC    STATUS, 33$                                     ;
                             52          62  D0 002D1        MOVL    (CURR_PP), CURR_PP                              ; 1480
                                         E2  11 002D4        BRB     30$                                             ; 1469
                      34    A6           08  88 002D6 31$:   BISB2   #8, 52(DCB)                                     ; 1489
```

```
                                                                           G 14
SMG$DISPLAY_LIN SMG$DISPLAY_LINKS - Virtual Display Linkages     16-Sep-1984 00:29:22    VAX-11 Bliss-32 V4.0-742            Page  36
1-096           SMG$CHANGE_VIRTUAL_DISPLAY - Change Virtual Dis 14-Sep-1984 13:09:43    [SMGRTL.SRC]SMGDISLIN.B32;1                (7)

                               50            01  D0 002DA 32$:     MOVL    #1, R0                                              ; 1492
                                                 04 002DD 33$:     RET                                                         ; 1493

; Routine Size:  734 bytes,    Routine Base:  _SMG$CODE + 0351


; 1239          1494  1 !<BLF/PAGE>
```

```
%SBTTL 'SMG$CHECK_FOR_OCCLUSION - Check to see if display is occluded'
GLOBAL ROUTINE SMG$CHECK_FOR_OCCLUSION (
                                        DISPLAY_ID,
                                        PASTEBOARD_ID,
                                        OCCLUSION_STATE
                                       ) =
!++
! FUNCTIONAL DESCRIPTION:
!
!       This procedure determines if the given virtual display, as
!       pasted to the given pasteboard, is occluded by another virtual
!       display.  The OCCLUSION state is set to:
!
!                         1        : if virtual display is occluded
!                         0        : if virtual display is not occluded.
!               not meaningfull : if status is not SS$_NORMAL.
!
!       The returned status reflects whether the question could be
!       answered at all.
!
! CALLING SEQUENCE:
!
!       ret_status.wlc.v = SMG$CHECK_FOR_OCCLUSION (
!                                       DISPLAY_ID.rl.r,
!                                       PASTEBOARD_ID.rl.r,
!                                       OCCLUSION_STATE.wl.r)
! FORMAL PARAMETERS:
!
!       DISPLAY_ID.rl.r         Address of a display id.
!
!       PASTEBOARD_ID.rl.r      Address of a pasteboard id.
!
!       OCCLUSION_STATE.wl.r    Set to          1  if occluded
!                                               0  if not occluded
!
!
! IMPLICIT INPUTS:
!
!       NONE
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! COMPLETION STATUS:
!
!       SS$_NORMAL       Normal success.  OCCLUSION_STATE calculated.
!
!       SMG$_NOTPASTED  Given virtual display is not pasted to given
!                        pasteboard.
!
!       SMG$_INVPAS_ID  Invalid pasteboard id.
!
!       SMG$_INVDIS_ID  Invalid display id.
!
! SIDE EFFECTS:
!
```

```
; 1298          1552  1 !       NONE
; 1299          1553  1 !--
; 1300          1554  1
; 1301          1555  2     BEGIN
; 1302          1556  2     LOCAL
; 1303          1557  2         STATUS,                 ! Status of subroutine calls
; 1304          1558  2
; 1305          1559  2         PP : REF $PP_DECL,      ! Address of relevant pasting packet
; 1306          1560  2
; 1307          1561  2         PBCB: REF $PBCB_DECL,   ! Address of Pasteboard Control Block
; 1308          1562  2
; 1309          1563  2         DCB : REF $DCB_DECL;    ! Address of Display Control Block
; 1310          1564  2
; 1311          1565  2 !+
; 1312          1566  2 ! Validate number of arguments.
; 1313          1567  2 !-
; 1314          1568  2     $SMG$VALIDATE_ARGCOUNT (3, 3);
; 1315          1569  2
; 1316          1570  2 !+
; 1317          1571  2 ! Get DCB and PBCB addresses that go with these display ids and
; 1318          1572  2 ! pasteboard ids.
; 1319          1573  2 !-
; 1320          1574  2     $SMG$GET_DCB (.DISPLAY_ID, DCB);
; 1321          1575  2     $SMG$GET_PBCB (.PASTEBOARD_ID, PBCB);
; 1322          1576  2
; 1323          1577  2 !+
; 1324          1578  2 ! Try to find the pasting packet that binds these two.  Return
; 1325          1579  2 ! SMG$_NOTPASTED it can't be located.
; 1326          1580  2 !-
; 1327          1581  3     IF NOT (STATUS = SMG$$LOCATE_PP ( .DCB, .PBCB, PP))
; 1328          1582  2     THEN
; 1329          1583  2         RETURN (.STATUS);
; 1330          1584  2
; 1331          1585  2 !+
; 1332          1586  2 ! Check to see if occluded and return appropriate OCCLUSION_STATE.
; 1333          1587  2 !-
; 1334          1588  3     .OCCLUSION_STATE = ( IF .PP [PP_V_OCCLUDED] THEN 1    ! Occluded
; 1335          1589  2                                                 ELSE 0);  ! Not occluded
; 1336          1590  2
; 1337          1591  2     RETURN SS$_NORMAL;
; 1338          1592  1     END;                    ! End of routine SMG$CHECK_FOR_OCCLUSION



                                  0004 00000          .ENTRY  SMG$CHECK_FOR_OCCLUSION, Save R2                        ; 1496
              52 00000000'    EF   9E 00002          MOVAB   PBD_L_COUNT, R2                                          ;
              5E              04   C2 00009          SUBL2   #4, SP                                                   ;
              03              6C   91 0000C          CMPB    (AP), #3                                                 ; 1568
                                   08   13 0000F          BEQL    1$                                                       ;
              50 00000000G    8F   D0 00011          MOVL    #SMG$_WRONUMARG, R0                                      ;
                                   04 00018          RET                                                              ;
              50         04   BC   D0 00019  1$:     MOVL    @DISPLAY_ID, R0                                          ; 1574
      04      BC         38   A0   D1 0001D          CMPL    56(R0), @DISPLAY_ID                                      ;
                              06   12 00022          BNEQ    2$                                                       ;
              11         44   A0   91 00024          CMPB    68(R0), #17                                              ;
```

```
                                         08  13 00028           BEQL    3$                                    ;
                         50 00000000G    8F  D0 0002A 2$:       MOVL    #SMG$_INVDIS_ID, R0                   ;
                                             04 00031           RET                                           ;
                         51          04  BC  D0 00032 3$:       MOVL    @DISPLAY_ID, DCB                      ;
                         50          08  BC  D0 00036           MOVL    @PASTEBOARD_ID, R0                    ; 1575
                                         0A  19 0003A           BLSS    4$                                    ;
                         62              50  D1 0003C           CMPL    R0, PBD_L_COUNT                       ;
                                         05  14 0003F           BGTR    4$                                    ;
             08       44 A2              50  E0 00041           BBS     R0, PBD_V_PB_AVAIL, 5$                ;
                         50 00000000G    8F  D0 00046 4$:       MOVL    #SMG$_INVPAS_ID, R0                   ;
                                             04 0004D           RET                                           ;
                         50          04 A240 D0 0004E 5$:       MOVL    PBD_A_PBCB[R0], PBCB                  ;
                                   4001  8F  BB 00053           PUSHR   #^M<R0,SP>                            ; 1581
                                         51  DD 00057           PUSHL   DCB                                   ;
                    0000V CF             03  FB 00059           CALLS   #3, SMG$$LOCATE_PP                    ;
                         15              50  E9 0005E           BLBC    STATUS, 8$                            ;
                         50              6E  D0 00061           MOVL    PP, R0                                ; 1588
                         05          2A  A0  E9 00064           BLBC    42(R0), 6$                            ;
                         50              01  D0 00068           MOVL    #1, R0                                ;
                                         02  11 0006B           BRB     7$                                    ;
                                         50  D4 0006D 6$:       CLRL    R0                                    ;
                    0C BC                50  D0 0006F 7$:       MOVL    R0, @OCCLUSION_STATE                  ;
                         50              01  D0 00073           MOVL    #1, R0                                ; 1591
                                             04 00076 8$:       RET                                           ; 1592

; Routine Size:  119 bytes,    Routine Base:  _SMG$CODE + 062F


; 1339          1593  1 !<BLF/PAGE>
```

```
%SBTTL 'SMG$CREATE_PASTEBOARD - Create Pasteboard'
GLOBAL ROUTINE SMG$CREATE_PASTEBOARD (
                                NEW_PBID,
                                OUT_DEVICE,
                                PB_ROWS,
                                PB_COLS,
                                PRESERVE_SCREEN_FLAG
                                    ) =
!++
! FUNCTIONAL DESCRIPTION:
!
!       This routine creates a new pasteboard -- returning its assigned
!       pasteboard_id.  OUT_DEVICE is the device upon which this
!       pasteboard is to be written.  If not supplied, output will flow
!       to SYS$OUTPUT.
!
!       If PB_ROWS and/or PB_COLS are provided, they are filled in with
!       the number of rows and number of columns on the physical device.
!
!       If called upon to create a 2nd pasteboard on a device that
!       already has a pasteboard associated with it, we simply return
!       the id of the already-existing pasteboard and the qualified
!       success SMG$_PASALREXI.
!
! CALLING SEQUENCE:
!
!       ret_status.wlc.v = SMG$CREATE_PASTEBOARD (
!                               NEW_PBID.wl.r
!                               [,OUT_DEVICE.rt.dx]
!                               [,PB_ROWS.wl.r]
!                               [,PB_COLS.wl.r]
!                               [,PRESERVE_SCREEN_FLAG.rl.r])
!
! FORMAL PARAMETERS:
!
!       NEW_PBID.wl.r       Pasteboard id of newly-created pasteboard.
!
!       OUT_DEVICE.rt.dx            [Optional].  If supplied, this parameter
!                                   is the file specification or logical
!                                   name upon which the output associated
!                                   with this pasteboard will be written.
!                                   If omitted, output goes to SYS$OUTPUT.
!
!       PB_ROWS             [Optional].  If provided, it is filled in with
!                           the number of rows on the physical device.
!
!       PB_COLS             [Optional].  If provided, it is filled in with
!                           the number of columns on the physical device.
!
!       PRESERVE_SCREEN_FLAG        [Optional].  If provided, and if has a
!                                   value of 1, then the screen will not
!                                   be initially cleared.
!
!
! IMPLICIT INPUTS:
!
!       NONE
```

```
; 1398          1651  1 !
; 1399          1652  1 ! IMPLICIT OUTPUTS:
; 1400          1653  1 !
; 1401          1654  1 !       NONE
; 1402          1655  1 !
; 1403          1656  1 ! COMPLETION STATUS:
; 1404          1657  1 !
; 1405          1658  1 !       SS$_NORMAL      Normal successful completion
; 1406          1659  1 !       LIB$_INSVIRMEM  Insufficient virtual memory to allocate needed
; 1407          1660  1 !                       buffer.
; 1408          1661  1 !       SMG$_PASALREXI  Pasteboard already exists for this device
; 1409          1662  1 !       SMG$_WRONUMARG  Wrong number of arguments.
; 1410          1663  1 !
; 1411          1664  1 ! SIDE EFFECTS:
; 1412          1665  1 !
; 1413          1666  1 !       NONE
; 1414          1667  1 !--
```

```
2       BEGIN
2
2       BUILTIN
2           NULLPARAMETER;
2
2       LOCAL
2           FS_LEN : WORD INITIAL (0),      ! Length of filespec name to use.
2
2           FS_ADDR,                    ! Address of filespec name to use.
2
2           STATUS,                     ! Status of subroutine calls
2
2           TERM_TYPE,                  ! terminal type
2
2           CLEAR_FLAG,                 ! TRUE means clear screen
2
2           PBID,                       ! Id of pasteboard being created.
2
2           PBCB : REF $PBCB_DECL;      ! Address of pasteboard control block
2                                       ! being created.
2
2       EXTERNAL ROUTINE
2
2           SMG$$ERASE_PASTEBOARD,
2           SMG$$OUT_OF_BAND_HANDLER;
2
2       $SMG$VALIDATE_ARGCOUNT (1, 5);      ! Test for right no. of args
2
2       $SMG$GET_NEXT_PID ( PBID);  ! Allocate a new PBID
2
2   !+
2   ! Decide what output device is to receive the the output of this
2   ! pasteboard.
2   !-
2       FS_LEN  = %CHARCOUNT  ('SYS$OUTPUT');       ! Assume default
2       FS_ADDR = UPLIT (BYTE ('SYS$OUTPUT'));
2
2       IF NOT NULLPARAMETER (OUT_DEVICE)
2       THEN
3           BEGIN   ! User-supplied filespec
4           IF NOT (STATUS = LIB$ANALYZE_SDESC_R2 ( .OUT_DEVICE ;
4                                                   FS_LEN, FS_ADDR))
3           THEN
3               RETURN ( .STATUS);
2           END;    ! User-supplied filespec
2
2   !+
2   ! Create a PBCB.  Allocate buffers, etc.
2   ! Extract the necessary device attributes and store in PBCB.
2   !-
2       STATUS = SMG$$SETUP_TERMINAL_TYPE (
2                   .FS_ADDR,                   ! filespec addr
2                   .FS_LEN,                    ! Len of filespec
2                    TERM_TYPE,                 ! Gets terminal type
2                    PBCB);                     ! Address to receive address of PBCB
2
2       IF NOT .STATUS
```

```
2           THEN
3               BEGIN
3               PBD_V_PB_AVAIL [.PBID] = 0;      ! Release PBID number
4               RETURN (.STATUS)
2               END;
2
2       !+
2       ! Decide whether we want to handle this output device ourselves
2       ! or use RMS to handle it.  We use RMS if the output device is
2       ! not a terminal.  We also use RMS if the output device is a terminal,
2       ! but one we can't handle, such as a hardcopy terminal.
2       !-
2
2       PBCB [PBCB_V_RMS] = (.PBCB[PBCB_B_CLASS]   NEQ DC$_TERM ) OR
2                           (.PBCB[PBCB_B_DEVTYPE] EQL UNKNOWN  ) OR
2                           (.PBCB[PBCB_B_DEVTYPE] EQL HARDCOPY );
2
2       !+
2       ! Loop through all the pasteboards we currently have trying to find
2       ! one whose associated resultant name string is the same as the one we
2       ! just created.
2       ! If we can find one, we have just created a 2nd pasteboard for the same
2       ! physical device and we want to get rid of the pasteboard we just
2       ! created and return to the caller the id of the pasteboard that already
2       ! exists for this device.
2       ! We do this only if the output device is a terminal.
2       ! If the output device is a file, we assume that the user wants
2       ! to create a new file for each pasteboard he creates.
2       !-
2
2       IF NOT .PBCB [PBCB_V_RMS]
2       THEN
2           INCR I FROM 0 TO .PBD_L_COUNT -1
2           DO
3               BEGIN   ! Loop thru pasteboards
3               LOCAL
3                   SEARCH_PBCB : REF $PBCB_DECL;       ! Addr of pasteboard
3                                                       ! control blocks that
3                                                       ! we are inspecting.
3
3               IF (SEARCH_PBCB = .PBD_A_PBCB [.I]) NEQ 0
3               THEN
4                   BEGIN       ! A valid pasteboard address
4                   IF .SEARCH_PBCB [PBCB_W_DEVNAM_LEN] EQL
4                      .PBCB        [PBCB_W_DEVNAM_LEN]
4                   THEN
5                       BEGIN ! Lengths match
5                       IF CH$EQL ( .SEARCH_PBCB [PBCB_W_DEVNAM_LEN],    ! length
5                                    SEARCH_PBCB [PBCB_T_DEVNAM],        ! addr
5                                   .PBCB [PBCB_W_DEVNAM_LEN],           ! length
5                                    PBCB [PBCB_T_DEVNAM])               ! addr
5                       THEN
6                           BEGIN       ! Match found
6                           LOCAL
6                               STATUS;         ! Local status of subr. calls
6
6                           !+
6                                   ! This physical device already has a pasteboard
6                                   ! associated with it.
6                                   ! Get rid of the one we just created.
6                                   ! First return PBID number we consumed -- we won't
6                                   ! be using it.
6                                   !-
6                                   PBD_V_PB_AVAIL [.PBID] = 0;
6
6                                   !+
6                                   ! Second deallocate the WCB that got allocated.
6                                   !-
6                                   IF .PBCB [PBCB_A_WCB] NEQ 0
6                                   THEN
7                                       IF NOT (STATUS = SMG$$DEALLOCATE_WCB (
7                                                      .PBCB [PBCB_A_WCB]) )
6                                       THEN
6                                           RETURN (.STATUS);
6
6                                   !+
6                                   ! Next release output buffer.
6                                   !-
6                                   IF .PBCB [PBCB_A_OUTPUT_BUFFER] NEQ 0
6                                   THEN
7                                       IF NOT ( STATUS = LIB$FREE_VM (
7                                                %REF (.PBCB [PBCB_W_OUTPUT_BUFSIZ]),
7                                                PBCB [PBCB_A_OUTPUT_BUFFER]))
6                                       THEN
6                                           RETURN (.STATUS);
6
6                                   !+
6                                   ! Finally release the PBCB itself.
6                                   !-
6                                   IF NOT (STATUS = LIB$FREE_VM ( %REF (PBCB_K_SIZE),
7                                                                  PBCB))
6                                   THEN
6                                       RETURN (.STATUS);
6
6                                   !+
6                                   ! Return as an id the id of the one that already
6                                   ! exists.
6                                   !-
6                                   .NEW_PBID = .SEARCH_PBCB [PBCB_L_PBID];
6
6                                   !+
6                                   ! If caller requested number of rows and columns on
6                                   ! device, tell him.
6                                   !-
6                                   IF NOT NULLPARAMETER (PB_ROWS)
6                                   THEN .PB_ROWS = .SEARCH_PBCB [PBCB_B_ROWS];
6
6                                   IF NOT NULLPARAMETER (PB_COLS)
6                                   THEN .PB_COLS = .SEARCH_PBCB [PBCB_W_WIDTH];
6
6                                   RETURN ( SMG$_PASALREXI );
5                                   END;          ! Match found
4                               END;  ! Lengths match
3                           END;        ! A valid pasteboard address
```

```
        END;    ! Loop thru pasteboards

!+
! If we fall out of loop, none of our current pasteboards are pasted to
! the same device.  Continue with the creation process.
! Store pasteboard id in the PBCB itself.
!-
    PBCB [PBCB_L_PBID] = .PBID;

!+
! Store the original name (that the user specified) for this device
! in the PBCB.  This name may include a filename as well as a
! device name.
! First we allocate virtual memory for this buffer and
! then we store the length and address in the PBCB for future reference.
!-

    STATUS=LIB$GET_VM(%REF(.FS_LEN), PBCB[PBCB_A_OUTNAM]);
    IF NOT .STATUS THEN RETURN (.STATUS);
    PBCB[PBCB_W_OUTNAM_LEN]=.FS_LEN;
    CH$MOVE(.FS_LEN,.FS_ADDR,.PBCB[PBCB_A_OUTNAM]);

!+
! If device is a terminal, assign a channel to it.
! If the device is not a terminal, allocate a FAB and RAB
! and open the file for output using RMS.
!-

IF .PBCB[PBCB_V_RMS]
  THEN  BEGIN   ! use RMS to open output

        !+
        ! Allocate a FAB and RAB to be used to talk to this file.
        !-

        STATUS=LIB$GET_VM(%REF(FAB$C_BLN),PBCB[PBCB_A_FAB]);
        IF NOT .STATUS THEN RETURN .STATUS;

        STATUS=LIB$GET_VM(%REF(RAB$C_BLN),PBCB[PBCB_A_RAB]);
        IF NOT .STATUS THEN RETURN .STATUS;

        !+
        ! Allocate a record buffer.
        ! This will be one byte larger than the width of
        ! the pasteboard because sometimes we will prepend
        ! a formfeed to the record.
        !-

        STATUS=LIB$GET_VM(%REF(.PBCB[PBCB_W_WIDTH]+1),PBCB[PBCB_A_RBF]);
        IF NOT .STATUS THEN RETURN .STATUS;

        !+
        ! Initialize the FAB and RAB.
        !-

        $FAB_INIT(      FAB     = .PBCB[PBCB_A_FAB],
                        DNM     = 'SMGOUTPUT.LIS',      ! default filename
; 1644        P 1896  3                       CT)     = .PBCB,          ! why not? pass the PBCB as user context
; 1645        P 1897  3                       FAC     = PUT,            ! write access only
; 1646        P 1898  3                       FNA     = .FS_ADDR,
; 1647        P 1899  3                       FNS     = .FS_LEN,
; 1648        P 1900  3                       ORG     = SEQ,            ! sequential file
; 1649        P 1901  3                       FOP     = SQO,            ! sequential operations only
; 1650        P 1902  3                       RAT     = CR,             ! carriage control
; 1651        P 1903  3                       RFM     = VAR,            ! variable length records
; 1652          1904  3                       MRS     = .PBCB[PBCB_W_WIDTH]+1); ! max record size
; 1653          1905  3
; 1654        P 1906  3           $RAB_INIT(  RAB     = .PBCB[PBCB_A_RAB],
; 1655        P 1907  3                       CTX     = .PBCB,          ! pass the PBCB as user context
; 1656        P 1908  3                       FAB     = .PBCB[PBCB_A_FAB],
; 1657        P 1909  3                       RBF     = .PBCB[PBCB_A_RBF],
; 1658          1910  3                       RAC     = SEQ);           ! sequential output
; 1659          1911  3
; 1660          1912  3           !+
; 1661          1913  3           ! Open the file for output.
; 1662          1914  3           !-
; 1663          1915  3
; 1664          1916  3           STATUS=$CREATE( FAB     = .PBCB[PBCB_A_FAB]);
; 1665          1917  3           IF NOT .STATUS THEN RETURN .STATUS;
; 1666          1918  3
; 1667          1919  3           !+
; 1668          1920  3           ! Connect a record stream to the file.
; 1669          1921  3           !-
; 1670          1922  3
; 1671          1923  3           STATUS=$CONNECT( RAB    = .PBCB[PBCB_A_RAB]);
; 1672          1924  3           IF NOT .STATUS THEN RETURN .STATUS;
; 1673          1925  3
; 1674          1926  3           END     ! use RMS to open output
; 1675          1927  3     ELSE  BEGIN   ! assigning channel
; 1676          1928  3
; 1677          1929  3           LOCAL   NAME_DESC       : VECTOR[2],      ! Fixed length descriptor
; 1678          1930  3                   ASYNC_EFN       : LONG,           ! Longword to hold efn
; 1679          1931  3                   TTIOSB          : VECTOR[4,WORD],! IOSB for SENSE MODE
; 1680          1932  3                   CHARBUF         : BLOCK[12,BYTE];! 12-byte characteristics buffer
; 1681          1933  3
; 1682          1934  3           !+
; 1683          1935  3           ! Create a fixed length descriptor for our device name string
; 1684          1936  3           ! for use by $ASSIGN.
; 1685          1937  3           !-
; 1686          1938  3
; 1687          1939  3           NAME_DESC[0]=.PBCB[PBCB_W_DEVNAM_LEN];
; 1688          1940  3           NAME_DESC[1]= PBCB[PBCB_T_DEVNAM];
; 1689          1941  3
; 1690          1942  3           !+
; 1691          1943  3           ! Assign the channel.
; 1692          1944  3           ! Put the resulting channel number in PBCB[PBCB_W_CHAN].
; 1693          1945  3           !-
; 1694          1946  3
; 1695        P 1947  3           STATUS=$ASSIGN( DEVNAM  = NAME_DESC,
; 1696          1948  3                           CHAN    = PBCB[PBCB_W_CHAN]);
; 1697          1949  3           IF NOT .STATUS THEN RETURN .STATUS;
; 1698          1950  3
; 1699          1951  3           !+
; 1700          1952  3           ! Assign an asynchronous event flag.
```

```
    3           !-
    3
    3           STATUS=LIB$GET_EF(ASYNC_EFN);
    3           IF NOT .STATUS THEN RETURN .STATUS;
    3
    3           !+
    3           ! Store the value into a byte in the PBCB.
    3           !-
    3
    3           PBCB [PBCB_B_ASYNC_EFN] = .ASYNC_EFN;
    3
    3           !+
    3           ! Do a SENSE MODE QIO to get additional characteristics
    3           ! of interest.
    3           ! Ignore everything returned in the characteristics buffer.
    3           ! (We already got that stuff.)
    3           ! The I/O status block has neat things of interest.
    3           !-
    3
P   3           STATUS=$QIOW(   CHAN    = .PBCB[PBCB_W_CHAN],
P   3                           FUNC    = IO$_SENSEMODE,
P   3                           IOSB    = TTIOSB,
P   3                           P1      = CHARBUF,
    3                           P2      = 12);
    3           IF NOT .STATUS THEN RETURN .STATUS;
    3           IF NOT .TTIOSB[0] THEN RETURN .TTIOSB[0];
    3
    3           PBCB [PBCB_W_SPEED]  = .TTIOSB[1];
    3           PBCB [PBCB_W_FILL]   = .TTIOSB[2];
    3           PBCB [PBCB_B_PARITY] = .TTIOSB[3]
    3
    2           END;    ! assigning channel
    2
    2   !+
    2   ! Set up our exit block which is contained within the PBCB.
    2   ! This exit block is used to establish an exit handler for
    2   ! this terminal. When the exit handler is called,
    2   ! it will flush the output buffers.
    2   ! This guarantees that the user will see all his output even if
    2   ! his program exits and he doesn't manually flush the buffers.
    2   !-
    2
    2       PBCB [PBCB_A_EXIT_ADDR] = SMG$$PBCB_EXIT_HANDLER;
    2                                           ! Address of our exit handler
    2       PBCB [PBCB_B_EXIT_ARGCNT] = 2;      ! Our exit handler gets called with
    2                                           ! two arguments.
    2       PBCB [PBCB_A_EXIT_RSN] = PBCB [PBCB_L_EXIT_REASON];
    2                                           ! The first argument is the address
    2                                           ! of the longword to receive the
    2                                           ! exit reason.  This longword appears
    2                                           ! elsewhere in the PBCB (not in
    2                                           ! the exit block).
    2       PBCB [PBCB_A_EXIT_PBCB] = .PBCB;    ! The second argument is the address
    2                                           ! of this PBCB.  This is needed
    2                                           ! because there are many PBCBs and
    2                                           ! one exit routine serves them all.
    2                                           ! There is a separate exit block for
```

```
                                                    ! each pasteboard.

!+
! Establish the exit handler, using the exit block just created.
!-

    STATUS=$DCLEXH(DESBLK=PBCB [PBCB_R_EXIT_BLOCK]);
    IF NOT .STATUS THEN RETURN .STATUS;

!+
! Now we do an incredible strange thing.
! We build a 10-byte routine in the PBCB to service out-of-band ASTs.
! The routine has the form:
!
!       0000    entry mask
!         FA    CALLG
!         6C    (AP)
!         9F    absolute addressing
!    address    longword address of SMG$$OUT_OF_BAND_HANDLER
!         04    RET
!
! Symbolically, the routine looks as follows:
!
!    ROUTINE BAND_HANDLER =
!       BEGIN
!       EXTERNAL ROUTINE SMG$$OUT_OF_BAND_HANDER : ADDRESSING_MODE(ABSOLUTE);
!       BUILTIN AP,CALLG;
!       RETURN CALLG(.AP,SMG$$OUT_OF_BAND_HANDLER);
!       END;
!
! However, we don't actually create this routine in BLISS and then
! move it into our structure, because we can't be guaranteed that
! BLISS will continue to generate the same code in future releases.
! Thus we create the entire routine ourselves.
! This code would have to change if we ever tried to run this
! on a machine with a new arhitecture.
!-

    PBCB[PBCB_W_ENTRY_MASK]     = %X'0000';
    PBCB[PBCB_B_CALLG]          = %X  'FA';
    PBCB[PBCB_B_REG_AP]         = %X  '6C';
    PBCB[PBCB_B_ABS]            = %X  '9F';
    PBCB[PBCB_A_BAND_HANDLER]   = SMG$$OUT_OF_BAND_HANDLER;
    PBCB[PBCB_B_RET]            = %X  '04';

!+
! Since all went well, we can now adjust the count of how many PBCB's
! we have and plug its address into the pasteboard directory.
!-
    PBD_L_COUNT = .PBD_L_COUNT + 1;

    PBD_A_PBCB [.PBID] = .PBCB;

!+
! Initially clear the screen (unless we are asked to preserve it).
!-
```

```
2          CLEAR_FLAG=1;
2          IF NOT NULLPARAMETER(PRESERVE_SCREEN_FLAG)
2          THEN  CLEAR_FLAG=NOT ..PRESERVE_SCREEN_FLAG;
2
2          IF .CLEAR_FLAG
2          THEN
3              BEGIN
3              STATUS=SMG$$ERASE_PASTEBOARD(.PBCB);
3              IF NOT .STATUS THEN RETURN .STATUS;
3              END
2          ELSE
3              BEGIN   ! Just pretend we cleared the screen.
3              LOCAL   WCB     : REF $WCB_DECL;
3              WCB=.PBCB[PBCB_A_WCB];
3
3              CH$FILL(%C' ',.WCB[WCB_L_BUFSIZE],.WCB[WCB_A_TEXT_BUF]);
3              CH$FILL(%C' ',.WCB[WCB_L_BUFSIZE],.WCB[WCB_A_SCR_TEXT_BUF]);
3              CH$FILL(0,    .WCB[WCB_L_BUFSIZE],.WCB[WCB_A_ATTR_BUF]);
3              CH$FILL(0,    .WCB[WCB_L_BUFSIZE],.WCB[WCB_A_SCR_ATTR_BUF]);
3
3              IF .WCB[WCB_A_CHAR_SET_BUF] NEQ 0
3              THEN
4                      BEGIN
4                      CH$FILL(0,.WCB[WCB_L_BUFSIZE],.WCB[WCB_A_CHAR_SET_BUF]);
3                      END;
3
3              IF .WCB[WCB_A_SCR_CHAR_SET_BUF] NEQ 0
3              THEN
4                      BEGIN
4                      CH$FILL(0,.WCB[WCB_L_BUFSIZE],.WCB[WCB_A_SCR_CHAR_SET_BUF]);
3                      END;
3
3              !+
3              ! The physical cursor moves to (1,1).
3              !-
3
3              ! WCB[WCB_W_CURR_CUR_ROW]=1;
3              ! WCB[WCB_W_OLD_CUR_ROW] =1;
3              ! WCB[WCB_W_CURR_CUR_COL]=1;
3              ! WCB[WCB_W_OLD_CUR_COL] =1;
3
3              !+
3              ! The line characteristics get set back to 0.
3              !-
3
3              CH$FILL(0,.WCB[WCB_W_NO_ROWS]+1,.WCB[WCB_A_LINE_CHAR]);
3              CH$FILL(0,.WCB[WCB_W_NO_ROWS]+1,.WCB[WCB_A_SCR_LINE_CHAR]);
3
2              END;    ! Just pretend we cleared the screen
2
2      !+
2      ! If caller is interested in number of rows and columns on device, tell
2      ! him.
2      !-
2          IF NOT NULLPARAMETER(PB_ROWS) THEN .PB_ROWS = .PBCB [PBCB_B_ROWS];
2          IF NOT NULLPARAMETER(PB_COLS) THEN .PB_COLS = .PBCB [PBCB_W_WIDTH];
2
```

```
; 1872         2124   2 !+
; 1873         2125   2 ! Return the new pasteboard id to caller
; 1874         2126   2 !-
; 1875         2127   2     .NEW_PBID = .PBID;
; 1876         2128   2
; 1877         2129   3     RETURN (SS$_NORMAL)
; 1878         2130   1     END;                            ! Routine SMG$CREATE_PASTEBOARD



                                                            006A6         .BLKB   2
                    54  55  50  54  55  4F  24  53  59  53  006A8 P.AAA:  .ASCII  \SYS$OUTPUT\                                 ;
        53  49  4C  2E  54  55  50  54  55  4F  47  4D  53  006B2 P.AAB:  .ASCII  \SMGOUTPUT.LIS\                              ;

                                                                          .EXTRN  SMG$$OUT_OF_BAND_HANDLER
                                                                          .EXTRN  SYS$CREATE, SYS$CONNECT
                                                                          .EXTRN  SYS$ASSIGN, SYS$QIOW
                                                                          .EXTRN  SYS$DCLEXH

                                                       0FFC 00000         .ENTRY  SMG$CREATE_PASTEBOARD, Save R2,R3,R4,R5,R6,-; 1595
                                                                                  R7,R8,R9,R10,R11                             ;
                                       5E            2C  C2 00002         SUBL2   #44, SP                                      ;
                                                     58  B4 00005         CLRW    FS_LEN                                       ; 1668
                         50            6C            01  83 00007         SUBB3   #1, (AP), DIFF                               ; 1694
                                       04            50  91 0000B         CMPB    DIFF, #4                                     ;
                                                     08  1B 0000E         BLEQU   1$                                           ;
                                       50 00000000G  8F  D0 00010         MOVL    #SMG$_WRONUMARG, R0                          ;
                                                         04 00017         RET                                                  ;
                         50 00000000'  EF            01  C1 00018 1$:     ADDL3   #1, PBD_L_COUNT, R0                          ; 1696
                                       10            50  D1 00020         CMPL    R0, #16                                      ;
                                                     08  15 00023         BLEQ    2$                                           ;
                                       50 00000000G  8F  D0 00025         MOVL    #SMG$_TOOMANPAS, R0                          ;
                                                         04 0002C         RET                                                  ;
           5A 00000000'  EF 00000000'  EF 00000000'  EF  EB 0002D 2$:     FFC     ZERO, PBD_K_MAX_PB_BY_REF, PBD_V_PB_AVAIL, - ;
                                                                                  PBID                                         ;
                         00 00000000'  EF            5A  E2 0003E         BBSS    PBID, PBD_V_PB_AVAIL, 3$                     ;
                                       58            0A  B0 00046 3$:     MOVW    #10, FS_LEN                                  ; 1702
                                       5B       9D   AF  9E 00049         MOVAB   P.AAA, FS_ADDR                               ; 1703
                                       02            6C  91 0004D         CMPB    (AP), #2                                     ; 1705
                                                     1B  1F 00050         BLSSU   4$                                           ;
                                                08   AC  D5 00052         TSTL    8(AP)                                        ;
                                                     16  13 00055         BEQL    4$                                           ;
                                       50       08   AC  D0 00057         MOVL    OUT_DEVICE, R0                               ; 1708
                                          00000000G  00  16 0005B         JSB     LIB$ANALYZE_SDESC_R2                         ;
                                       59            50  D0 00061         MOVL    R0, STATUS                                   ;
                                       5B            52  D0 00064         MOVL    R2, R11                                      ;
                                       58            51  D0 00067         MOVL    R1, FS_LEN                                   ;
                                       20            59  E9 0006A         BLBC    STATUS, 5$                                   ;
                                                08   AE  9F 0006D 4$:     PUSHAB  PBCB                                         ; 1718
                                                08   AE  9F 00070         PUSHAB  TERM_TYPE                                    ;
                                       7E            58  3C 00073         MOVZWL  FS_LEN, -(SP)                                ; 1720
                                                     5B  DD 00076         PUSHL   FS_ADDR                                      ; 1719
                            00000000G  00            04  FB 00078         CALLS   #4, SMG$$SETUP_TERMINAL_TYPE                 ;
                                       59            50  D0 0007F         MOVL    R0, STATUS                                   ;
                                       0B            59  E8 00082         BLBS    STATUS, 6$                                   ; 1724
                         00 00000000'  EF            5A  E5 00085         BBCC    PBID, PBD_V_PB_AVAIL, 5$                     ; 1727
                                                   D2CC  31 0008D 5$:     BRW     31$                                          ; 1728
```

```
                                                                  I 15
SMG$DISPLAY_LIN SMG$DISPLAY_LINKS - Virtual Display Linkages    16-Sep-1984 00:29:22    VAX-11 Bliss-32 V4.0-742               Page  51
1-096           SMG$CREATE_PASTEBOARD - Create Pasteboard       14-Sep-1984 13:09:43    [SMGRTL.SRC]SMGDISLIN.B32;1                 (10)
                                         50        08   AE  D0 00090 6$:    MOVL     PBCB, R0                                    ; 1738
                                                        51  D4 00094        CLRL     R1                                          ;
                                    42   8F        58   A0  91 00096        CMPB     88(R0), #66                                 ;
                                                        02  13 0009B        BEQL     7$                                          ;
                                                        51  D6 0009D        INCL     R1                                          ;
                                                        52  D4 0009F 7$:    CLRL     R2                                          ; 1739
                                                   10   A0  95 000A1        TSTB     16(R0)                                      ;
                                                        02  12 000A4        BNEQ     8$                                          ;
                                                        52  D6 000A6        INCL     R2                                          ;
                                         52             51  C8 000A8 8$:    BISL2    R1, R2                                      ;
                                                        51  D4 000AB        CLRL     R1                                          ; 1740
                                         05        10   A0  91 000AD        CMPB     16(R0), #5                                  ;
                                                        02  12 000B1        BNEQ     9$                                          ;
                                                        51  D6 000B3        INCL     R1                                          ;
                          53             51             52  89 000B5 9$:    BISB3    R2, R1, R3                                  ;
    00D0    C0            01             03             53  F0 000B9        INSV     R3, #3, #1, 208(R0)                         ;
                          03       00D0  C0             03  E1 000C0        BBC      #3, 208(R0), 10$                            ; 1755
                                                      00AB  31 000C6        BRW      22$                                         ;
                                         57 00000000'   EF  D0 000C9 10$:   MOVL     PBD_L_COUNT, R7                             ; 1757
                                         56             01  CE 000D0        MNEGL    #1, I                                       ;
                                                      0095  31 000D3 11$:   BRW      20$                                         ;
                                         55 00000000'EF46   D0 000D6 12$:   MOVL     PBD_A_PBCB[I], SEARCH_PBCB                  ; 1765
                                                        F3  13 000DE        BEQL     11$                                         ;
                                         54        08   AE  D0 000E0        MOVL     PBCB, R4                                    ; 1769
                                    12   A4        12   A5  B1 000E4        CMPW     18(SEARCH_PBCB), 18(R4)                     ;
                                                        E8  12 000E9        BNEQ     11$                                         ;
      12    A4            00        18   A5        12   A5  2D 000EB        CMPC5    18(SEARCH_PBCB), 24(SEARCH_PBCB), #0, -     ; 1775
                                                   18   A4     000F3                 18(R4), 24(R4)                              ;
                                                        74  12 000F5        BNEQ     20$                                         ;
                          00 00000000'   EF             5A  E5 000F7        BBCC     PBID, PBD_V_PB_AVAIL, 13$                   ; 1788
                                                   08   A4  D5 000FF 13$:   TSTL     8(R4)                                       ; 1793
                                                        0B  13 00102        BEQL     14$                                         ;
                                                   08   A4  DD 00104        PUSHL    8(R4)                                       ; 1796
                                 0000V   CF             01  FB 00107        CALLS    #1, SMG$$DEALLOCATE_WCB                     ;
                                         2D             50  E9 0010C        BLBC     STATUS, 16$                                 ; 1795
                                                   6C   A4  D5 0010F 14$:   TSTL     108(R4)                                     ; 1803
                                                        15  13 00112        BEQL     15$                                         ;
                                                   6C   A4  9F 00114        PUSHAB   108(R4)                                     ; 1807
                                    04   AE        70   A4  3C 00117        MOVZWL   112(R4), 4(SP)                              ; 1806
                                                   04   AE  9F 0011C        PUSHAB   4(SP)                                       ;
                             00000000G   00             02  FB 0011F        CALLS    #2, LIB$FREE_VM                             ; 1807
                                         13             50  E9 00126        BLBC     STATUS, 16$                                 ;
                                                   08   AE  9F 00129 15$:   PUSHAB   PBCB                                        ; 1814
                                    04   AE      014C   8F  3C 0012C        MOVZWL   #332, 4(SP)                                 ;
                                                   04   AE  9F 00132        PUSHAB   4(SP)                                       ;
                             00000000G   00             02  FB 00135        CALLS    #2, LIB$FREE_VM                             ;
                                         01             50  E8 0013C 16$:   BLBS     STATUS, 17$                                 ;
                                                            04 0013F        RET                                                  ;
                                    04   BC        14   A5  D0 00140 17$:   MOVL     20(SEARCH_PBCB), @NEW_PBID                  ; 1823
                                         03             6C  91 00145        CMPB     (AP), #3                                    ; 1829
                                                        0A  1F 00148        BLSSU    18$                                         ;
                                                   0C   AC  D5 0014A        TSTL     12(AP)                                      ;
                                                        05  13 0014D        BEQL     18$                                         ;
                                    0C   BC        5F   A5  9A 0014F        MOVZBL   95(SEARCH_PBCB), @PB_ROWS                   ; 1830
                                         04             6C  91 00154 18$:   CMPB     (AP), #4                                    ; 1832
                                                        0A  1F 00157        BLSSU    19$                                         ;
                                                   10   AC  D5 00159        TSTL     16(AP)                                      ;
```

```
                                                               J 15
SMG$DISPLAY_LIN SMG$DISPLAY_LINKS - Virtual Display Linkages     16-Sep-1984 00:29:22     VAX-11 Bliss-32 V4.0-742            Page  52
1-096           SMG$CREATE_PASTEBOARD - Create Pasteboard        14-Sep-1984 13:09:43     [SMGRTL.SRC]SMGDISLIN.B32;1                (10)

                                         05  13 0015C          BEQL    19$                                     ;
                      10  BC        5A   A5  3C 0015E          MOVZWL  90(SEARCH_PBCB), @PB_COLS               ; 1833
                          50 00000000G   8F  D0 00163  19$:    MOVL    #SMG$_PASALREXI, R0                     ; 1835
                                             04 0016A          RET                                             ;
                02        56             57  F2 0016B  20$:    AOBLSS  R7, I, 21$                              ; 1757
                                         03  11 0016F          BRB     22$                                     ;
                                       FF62  31 00171  21$:    BRW     12$                                     ;
                          57        08   AE  D0 00174  22$:    MOVL    PBCB, R7                                ; 1846
                      14  A7             5A  D0 00178          MOVL    PBID, 20(R7)                            ;
                                  00B0   C7  9F 0017C          PUSHAB  176(R7)                                 ; 1856
                      04  AE             58  3C 00180          MOVZWL  FS_LEN, 4(SP)                           ;
                                    04   AE  9F 00184          PUSHAB  4(SP)                                   ;
               00000000G  00             02  FB 00187          CALLS   #2, LIB$GET_VM                          ;
                          59             50  D0 0018E          MOVL    R0, STATUS                              ;
                          5F             59  E9 00191          BLBC    STATUS, 24$                             ; 1857
                    00E4  C7             58  B0 00194          MOVW    FS_LEN, 228(R7)                         ; 1858
          00B0  D7        6B             58  28 00199          MOVC3   FS_LEN, (FS_ADDR), @176(R7)             ; 1859
                03  00D0  C7             03  E0 0019F          BBS     #3, 208(R7), 23$                        ; 1867
                                       00D3  31 001A5          BRW     27$                                     ;
                                  00E8   C7  9F 001A8  23$:    PUSHAB  232(R7)                                 ; 1874
                      04  AE        50   8F  9A 001AC          MOVZBL  #80, 4(SP)                              ;
                                    04   AE  9F 001B1          PUSHAB  4(SP)                                   ;
               00000000G  00             02  FB 001B4          CALLS   #2, LIB$GET_VM                          ;
                          59             50  D0 001BB          MOVL    R0, STATUS                              ;
                          32             59  E9 001BE          BLBC    STATUS, 24$                             ; 1875
                                  00EC   C7  9F 001C1          PUSHAB  236(R7)                                 ; 1877
                      04  AE        44   8F  9A 001C5          MOVZBL  #68, 4(SP)                              ;
                                    04   AE  9F 001CA          PUSHAB  4(SP)                                   ;
               00000000G  00             02  FB 001CD          CALLS   #2, LIB$GET_VM                          ;
                          59             50  D0 001D4          MOVL    R0, STATUS                              ;
                          19             59  E9 001D7          BLBC    STATUS, 24$                             ; 1878
                                  00F0   C7  9F 001DA          PUSHAB  240(R7)                                 ; 1887
                      04  AE        5A   A7  3C 001DE          MOVZWL  90(R7), 4(SP)                           ;
                                    04   AE  D6 001E3          INCL    4(SP)                                   ;
                                    04   AE  9F 001E6          PUSHAB  4(SP)                                   ;
               00000000G  00             02  FB 001E9          CALLS   #2, LIB$GET_VM                          ;
                          59             50  D0 001F0          MOVL    R0, STATUS                              ;
                          70             59  E9 001F3  24$:    BLBC    STATUS, 25$                             ; 1888
                          56      00E8   C7  D0 001F6          MOVL    232(R7), R6                             ; 1904
   0050   8F        00    6E             00  2C 001FB          MOVC5   #0, (SP), #0, #80, (R6)                 ;
                                         66     00202                                                          ;
                          66      5003   8F  B0 00203          MOVW    #20483, (R6)                            ;
                      04  A6        40   8F  9A 00208          MOVZBL  #64, 4(R6)                              ;
                      16  A6             01  90 0020D          MOVB    #1, 22(R6)                              ;
                      18  A6             57  D0 00211          MOVL    R7, 24(R6)                              ;
                      1D  A6      0200   8F  B0 00215          MOVW    #512, 29(R6)                            ;
                      1F  A6             02  90 0021B          MOVB    #2, 31(R6)                              ;
                      2C  A6             5B  D0 0021F          MOVL    FS_ADDR, 44(R6)                         ;
                      30  A6      FDCC   CF  9E 00223          MOVAB   P.AAB, 48(R6)                           ;
                      34  A6             58  90 00229          MOVB    FS_LEN, 52(R6)                          ;
                      35  A6             0D  90 0022D          MOVB    #13, 53(R6)                             ;
              36  A6  5A  A7             01  A1 00231          ADDW3   #1, 90(R7), 54(R6)                      ;
                          58      00EC   C7  D0 00237          MOVL    236(R7), R8                             ; 1910
   0044   8F        00    6E             00  2C 0023C          MOVC5   #0, (SP), #0, #68, (R8)                 ;
                                         68     00243                                                          ;
                          68      4401   8F  B0 00244          MOVW    #17409, (R8)                            ;
                      18  A8             57  D0 00249          MOVL    R7, 24(R8)                              ;
```

```
                                1E     A8  94 0024D          CLRB     30(R8)                                     ;
                28    A8      00F0     C7  D0 00250          MOVL     240(R7), 40(R8)                            ;
                3C    A8               56  D0 00256          MOVL     R6, 60(R8)                                 ;
                                       56  DD 0025A          PUSHL    R6                                         ; 1916
         00000000G    00               01  FB 0025C          CALLS    #1, SYS$CREATE                             ;
                      59               50  D0 00263          MOVL     R0, STATUS                                 ;
                      0F               59  E9 00266 25$:     BLBC     STATUS, 26$                                ; 1917
                                       58  DD 00269          PUSHL    R8                                         ; 1923
         00000000G    00               01  FB 0026B          CALLS    #1, SYS$CONNECT                            ;
                      59               50  D0 00272          MOVL     R0, STATUS                                 ;
                      6E               59  E8 00275          BLBS     STATUS, 29$                                ; 1924
                                  00E1 31  00278 26$:        BRW      31$                                        ;
                24    AE        12     A7  3C 0027B 27$:     MOVZWL   18(R7), NAME_DESC                          ; 1939
                28    AE        18     A7  9E 00280          MOVAB    24(R7), NAME_DESC+4                        ; 1940
                                       7E  7C 00285          CLRQ     -(SP)                                      ; 1948
                                64     A7  9F 00287          PUSHAB   100(R7)                                    ;
                                30     AE  9F 0028A          PUSHAB   NAME_DESC                                  ;
         00000000G    00               04  FB 0028D          CALLS    #4, SYS$ASSIGN                             ;
                      59               50  D0 00294          MOVL     R0, STATUS                                 ;
                      DE               59  E9 00297          BLBC     STATUS, 26$                                ; 1949
                                0C     AE  9F 0029A          PUSHAB   ASYNC_EFN                                  ; 1955
         00000000G    00               01  FB 0029D          CALLS    #1, LIB$GET_EF                             ;
                      59               50  D0 002A4          MOVL     R0, STATUS                                 ;
                      CE               59  E9 002A7          BLBC     STATUS, 26$                                ; 1956
                67    A7        0C     AE  90 002AA          MOVB     ASYNC_EFN, 103(R7)                         ; 1962
                                       7E  7C 002AF          CLRQ     -(SP)                                      ; 1976
                                       7E  7C 002B1          CLRQ     -(SP)                                      ;
                                       0C  DD 002B3          PUSHL    #12                                        ;
                                24     AE  9F 002B5          PUSHAB   CHARBUF                                    ;
                                       7E  7C 002B8          CLRQ     -(SP)                                      ;
                                3C     AE  9F 002BA          PUSHAB   TTIOSB                                     ;
                                       27  DD 002BD          PUSHL    #39                                        ;
                      7E        64     A7  3C 002BF          MOVZWL   100(R7), -(SP)                             ;
                                       7E  D4 002C3          CLRL     -(SP)                                      ;
         00000000G    00               0C  FB 002C5          CALLS    #12, SYS$QIOW                              ;
                      59               50  D0 002CC          MOVL     R0, STATUS                                 ;
                      A6               59  E9 002CF          BLBC     STATUS, 26$                                ; 1977
                      05        1C     AE  E8 002D2          BLBS     TTIOSB, 28$                                ; 1978
                      50        1C     AE  3C 002D6          MOVZWL   TTIOSB, R0                                 ;
                                       04  002DA             RET                                                 ;
              00B4    C7        1E     AE  D0 002DB 28$:     MOVL     TTIOSB+2, 180(R7)                          ; 1980
                11    A7        22     AE  90 002E1          MOVB     TTIOSB+6, 17(R7)                           ; 1982
                78    A7 00000000G     00  9E 002E6 29$:     MOVAB    SMG$$PBCB_EXIT_HANDLER, 120(R7)            ; 1995
                7C    A7               02  90 002EE          MOVB     #2, 124(R7)                                ; 1997
              0080    C7      0088     C7  9E 002F2          MOVAB    136(R7), 128(R7)                           ; 1999
              0084    C7               57  D0 002F9          MOVL     R7, 132(R7)                                ; 2005
                                74     A7  9F 002FE          PUSHAB   116(R7)                                    ; 2016
         00000000G    00               01  FB 00301          CALLS    #1, SYS$DCLEXH                             ;
                      59               50  D0 00308          MOVL     R0, STATUS                                 ;
                      4E               59  E9 0030B          BLBC     STATUS, 31$                                ; 2017
              008C    C7 6CFA0000      8F  D0 0030E          MOVL     #1828323328, 140(R7)                       ; 2048
              0090    C7        9F     8F  90 00317          MOVB     #-97, 144(R7)                              ; 2051
              0091    C7 00000000G     00  9E 0031D          MOVAB    SMG$$OUT_OF_BAND_HANDLER, 145(R7)          ; 2052
              0095    C7               04  90 00326          MOVB     #4, 149(R7)                                ; 2053
                         00000000'     EF  D6 0032B          INCL     PBD_L_COUNT                                ; 2059
         00000000'EF4A                 57  D0 00331          MOVL     R7, PBD_A_PBCB[PBID]                       ; 2061
                      50               01  D0 00339          MOVL     #1, CLEAR_FLAG                             ; 2067
```

```
                                                                 L 15
SMG$DISPLAY_LIN SMG$DISPLAY_LINKS - Virtual Display Linkages     16-Sep-1984 00:29:22    VAX-11 Bliss-32 V4.0-742          Page  54
1-096           SMG$CREATE_PASTEBOARD - Create Pasteboard        14-Sep-1984 13:09:43    [SMGRTL.SRC]SMGDISLIN.B32;1            (10)

                                        05          6C  91 0033C          CMPB    (AP), #5                                 ; 2068
                                        09          1F  0033F             BLSSU   30$                                      ;
                                              14    AC  D5 00341          TSTL    20(AP)                                   ;
                                        04          13  00344             BEQL    30$                                      ;
                                        50    14    BC  D2 00346          MCOML   @PRESERVE_SCREEN_FLAG, CLEAR_FLAG        ; 2069
                                        13          50  E9 0034A 30$:     BLBC    CLEAR_FLAG, 32$                          ; 2071
                                                    57  DD 0034D          PUSHL   R7                                       ; 2074
                             00000000G  00          01  FB 0034F          CALLS   #1, SMG$$ERASE_PASTEBOARD                ;
                                        59          50  D0 00356          MOVL    R0, STATUS                               ;
                                        54          59  E8 00359          BLBS    STATUS, 35$                              ; 2075
                                        50          59  D0 0035C 31$:     MOVL    STATUS, R0                               ;
                                                        04 0035F          RET                                              ;
                                        59    08    A7  D0 00360 32$:     MOVL    8(R7), WCB                               ; 2080
                                        56    28    A9  D0 00364          MOVL    40(WCB), R6                              ; 2082
        56           20                 6E          00  2C 00368          MOVC5   #0, (SP), #32, R6, @8(WCB)               ;
                                              08    B9     0036D                                                           ;
        56           20                 6E          00  2C 0036F          MOVC5   #0, (SP), #32, R6, @20(WCB)              ; 2083
                                              14    B9     00374                                                           ;
        56           00                 6E          00  2C 00376          MOVC5   #0, (SP), #0, R6, @12(WCB)               ; 2084
                                              0C    B9     0037B                                                           ;
        56           00                 6E          00  2C 0037D          MOVC5   #0, (SP), #0, R6, @24(WCB)               ; 2085
                                              18    B9     00382                                                           ;
                                              10    A9  D5 00384          TSTL    16(WCB)                                  ; 2087
                                                    07  13 00387          BEQL    33$                                      ;
        56           00                 6E          00  2C 00389          MOVC5   #0, (SP), #0, R6, @16(WCB)               ; 2090
                                              10    B9     0038E                                                           ;
                                              1C    A9  D5 00390 33$:     TSTL    28(WCB)                                  ; 2093
                                                    07  13 00393          BEQL    34$                                      ;
        56           00                 6E          00  2C 00395          MOVC5   #0, (SP), #0, R6, @28(WCB)               ; 2096
                                              1C    B9     0039A                                                           ;
                                        56    02    A9  3C 0039C 34$:     MOVZWL  2(WCB), R6                               ; 2112
                                                    56  D6 003A0          INCL    R6                                       ;
        56           00                 6E          00  2C 003A2          MOVC5   #0, (SP), #0, R6, @44(WCB)               ;
                                              2C    B9     003A7                                                           ;
        56           00                 6E          00  2C 003A9          MOVC5   #0, (SP), #0, R6, @48(WCB)               ; 2113
                                              30    B9     003AE                                                           ;
                                        03          6C  91 003B0 35$:     CMPB    (AP), #3                                 ; 2121
                                                    0A  1F 003B3          BLSSU   36$                                      ;
                                              0C    AC  D5 003B5          TSTL    12(AP)                                   ;
                                                    05  13 003B8          BEQL    36$                                      ;
                                  0C    BC    5F    A7  9A 003BA          MOVZBL  95(R7), @PB_ROWS                         ;
                                        04          6C  91 003BF 36$:     CMPB    (AP), #4                                 ; 2122
                                                    0A  1F 003C2          BLSSU   37$                                      ;
                                              10    AC  D5 003C4          TSTL    16(AP)                                   ;
                                                    05  13 003C7          BEQL    37$                                      ;
                                  10    BC    5A    A7  3C 003C9          MOVZWL  90(R7), @PB_COLS                         ;
                                  04    BC          5A  D0 003CE 37$:     MOVL    PBID, @NEW_PBID                          ; 2127
                                        50          01  D0 003D2          MOVL    #1, R0                                   ; 2129
                                                        04 003D5          RET                                              ; 2130

; Routine Size:  982 bytes,    Routine Base:  _SMG$CODE + 06BF


; 1879          2131  1 !<BLF/PAGE>
```

```
1 %SBTTL 'SMG$DELETE_PASTEBOARD - Delete Pasteboard'
1 GLOBAL ROUTINE SMG$DELETE_PASTEBOARD ( PBID, CLEAR_SCREEN_FLAG ) =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine terminates all use of a given physical display.
1 !       It deallocates the pasteboard control block and all its
1 !       substructures.  It gets rid of the event flag and the channel
1 !       number.  It removes any associated exit handler.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       ret_status.wlc.v = SMG$DELETE_PASTEBOARD ( PBID.rl.r
1 !                                                 [,CLEAR_SCREEN_FLAG.rl.r])
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       PBID.rl.r                Pasteboard id of pasteboard.
1 !
1 !       CLEAR_SCREEN_FLAG.rl.r  Set to 1 to clear the screen,
1 !                               0 to keep it as is.
1 !                               The default is to clear the screen.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !       SS$_NORMAL      Normal successful completion
1 !       SMG$_WRONUMARG  Wrong number of arguments.
1 !       SS$_xyz         errors from $DASSGN
1 !       LIB$_xyz        errors from LIB$FREE_VM or LIB$FREE_EF
1 !       SMG$_xyz        errors from SMG$$FLUSH_BUFFER
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !--
```

```
2 BEGIN
2
2 BUILTIN
2         NULLPARAMETER;
2
2 LOCAL
2
2         STATUS,                          ! Status of subroutine calls
2
2         CURR_PP : REF $PP_DECL,          ! Pasting packet pointer
2
2         WCB     : REF $WCB_DECL,         ! Window control block.
2
2         PBCB    : REF $PBCB_DECL;        ! Address of pasteboard control
2                                          ! block
2
2 EXTERNAL ROUTINE
2
2         SMG$$FORCE_SCROLL_REG,
2         SMG$$ERASE_PASTEBOARD,
2         SMG$$FLUSH_BUFFER,
2         SMG$CHANGE_PBD_CHARACTERISTICS;
2
2 $SMG$VALIDATE_ARGCOUNT (1, 2);  ! Test for right no. of args
2
2 $SMG$GET_PBCB (.PBID,PBCB);     ! Get address of PBCB
2
2 !+
2 ! Batch up the unpastes, so that the whole screen disappears at once.
2 !-
3     IF NOT (STATUS = SMG$$BEGIN_PASTEBOARD_UPDATE_R1(.PBCB))
2     THEN
2         RETURN (.STATUS);
2
2 !+
2 ! Walk chain of all DCB's pasted to this pasteboard and unpaste each.
2 !-
2     CURR_PP = .PBCB [PBCB_A_PP_PREV];
2     WHILE .CURR_PP NEQ PBCB [PBCB_A_PP_NEXT]
2     DO
3         BEGIN   ! Walk chain
3         LOCAL
3             DCB         : REF $DCB_DECL,        ! Address of DCB involved
3             PP_BASE     : REF $PP_DECL;         ! Base addr of this PP
3
3         PP_BASE = .CURR_PP - PP_PBCB_QUEUE_OFFSET;  ! Since queue header
3                                                     ! not at top of
3                                                     ! structure.
3         DCB = .PP_BASE [PP_A_DCB_ADDR];
4         IF NOT (STATUS = SMG$$UNPASTE_VIRTUAL_DISPLAY (
4                             .DCB,                   ! DCB involved
4                             .PBCB))                 ! PBCB involved
3         THEN
3             RETURN (.STATUS);
3
3         CURR_PP = .PP_BASE [PP_A_PREV_PBCB]; ! Step to next PP
2         END;    ! Walk chain
```

```
2
2       PBCB[PBCB_L_BATCH_LEVEL]=0;
2
2 !+
2 ! If the user asked for the screen to be erased, then
2 ! release lock on pasteboard, force output of now-blank screen, and
2 ! flush it out.
2 !-
2       IF NULLPARAMETER(CLEAR_SCREEN_FLAG)
3       OR (NOT NULLPARAMETER(CLEAR_SCREEN_FLAG) AND ..CLEAR_SCREEN_FLAG)
2       THEN
3           BEGIN   ! clear screen
3
3 !(b)      IF NOT (STATUS = SMG$$END_PASTEBOARD_UPDATE_R2(.PBCB))
3 !(b)      THEN
3 !(b)          RETURN (.STATUS);
3
3 !(b)      IF NOT ( STATUS = SMG$$CHECK_FOR_OUTPUT_PBCB(.PBCB))
3 !(b)      THEN
3 !(b)          RETURN (.STATUS);
3
3 !(b)      IF NOT (STATUS = SMG$$FLUSH_BUFFER(.PBCB))
3 !(b)      THEN
3 !(b)          RETURN (.STATUS);
3
3 ! Note (b): Erase pasteboard should clear the screen and
3 !           we can bypass flushing since the user is deleting his
3 !           pasteboard anyhow.
3
3           PBCB[PBCB_V_BUF_ENABLED]=0;
3
4           IF NOT (STATUS = SMG$$ERASE_PASTEBOARD(.PBCB))
3           THEN
3               RETURN (.STATUS);
3
3           !+
3           ! Set terminal back to it's orignal width.
3           ! This requires batching to be off.
3           !-
3
3           IF .PBCB[PBCB_W_WIDTH] NEQ .PBCB[PBCB_W_ORIG_WIDTH]
4             THEN  BEGIN
4                   STATUS=SMG$CHANGE_PBD_CHARACTERISTICS(.PBID,
4                                   %REF(.PBCB[PBCB_W_ORIG_WIDTH]));
4                   IF NOT .STATUS THEN RETURN .STATUS;
4                   STATUS=SMG$$FLUSH_BUFFER(.PBCB);
4                   IF NOT .STATUS THEN RETURN .STATUS
3                   END;
3
3           END     ! clear screen
2       ELSE
3           BEGIN
3           SMG$$FLUSH_BUFFER(.PBCB);
3           PBCB[PBCB_V_BUF_ENABLED]=0;
2           END;
2
2 WCB=.PBCB[PBCB_A_WCB];
```

```

!+
! If a scrolling region is set (other than the full screen),
! then reset it now, being careful to leave the cursor alone
! even though SET SCROLLING REGION may move it.
! Note that if we never established any scrolling regions,
! the TOP_SCROLL line will be 0.
!-

IF    .PBCB[PBCB_W_TOP_SCROLL_LINE] NEQ 0
AND  (.PBCB[PBCB_W_TOP_SCROLL_LINE] NEQ 1   OR
      .PBCB[PBCB_W_BOT_SCROLL_LINE] NEQ .WCB[WCB_W_NO_ROWS])
THEN
    BEGIN       ! Remove scrolling regions

    LOCAL

        FINAL_ROW,      ! Final cursor row
        FINAL_COL;      ! Final cursor column

    !+
    ! Construct escape sequence (possibly null if not a supporting terminal)
    ! to set the hardware scroll region to the full height of the screen.
    !-

    $SMG$GET_TERM_DATA(SET_SCROLL_REGION,
                        1,
                        .WCB [WCB_W_NO_ROWS]);

    !+
    ! Output BUFFER.
    !-

    IF .PBCB[PBCB_L_CAP_LENGTH] NEQ 0
    THEN
        BEGIN   ! Issue the reset

        !+
        ! Remember where the user left the physical cursor, since
        ! changing scrolling regions might upset this.
        !-

        FINAL_ROW=.WCB[WCB_W_CURR_CUR_ROW];
        FINAL_COL=.WCB[WCB_W_CURR_CUR_COL];

        STATUS = SMG$$OUTPUT(.PBCB,.PBCB[PBCB_L_CAP_LENGTH],
                                   .PBCB[PBCB_A_CAP_BUFFER]);
        IF NOT .STATUS THEN RETURN .STATUS;

        !+
        ! Move the cursor back to where it was.
        !-

        IF NOT NULLPARAMETER(CLEAR_SCREEN_FLAG)
        AND NOT ..CLEAR_SCREEN_FLAG
          THEN  BEGIN   ! Restore final cursor position

```

```
5                       $SMG$GET_TERM_DATA(SET_CURSOR_ABS,.FINAL_ROW,.FINAL_COL);
5
5                       STATUS = SMG$$OUTPUT(.PBCB,.PBCB[PBCB_L_CAP_LENGTH],
5                                             .PBCB[PBCB_A_CAP_BUFFER]);
5                       IF NOT .STATUS THEN RETURN .STATUS
5
5                       END     ! Restore final cursor position
5
4               END     ! Issue the reset
4
2           END;        ! Remove scrolling regions
2
2   !+
2   ! Get rid of our exit handler.  Ignore a no handler found error.
2   !-
2
2   STATUS=$CANEXH(DESBLK=PBCB[PBCB_R_EXIT_BLOCK]);
3   IF (NOT .STATUS) AND (.STATUS NEQ SS$_NOHANDLER)
2     THEN  RETURN  .STATUS;
2
2   !+
2   ! Deallocate the WCB if there is one.
2   !-
2
2   IF .PBCB[PBCB_A_WCB] NEQ 0
3     THEN  BEGIN   ! getting rid of WCB
3
3           STATUS=SMG$$DEALLOCATE_WCB(.PBCB[PBCB_A_WCB]);
3           PBCB[PBCB_A_WCB]=0;                             ! safety
3           IF NOT .STATUS THEN RETURN .STATUS
3
2           END;    ! getting rid of WCB
2
2   !+
2   ! If there is a channel assigned, deassign it now.
2   ! This automatically cancels any I/O on the channel.
2   ! In particular, it removes any out-of-band ASTs that
2   ! were enabled.
2   !-
2
2   IF .PBCB[PBCB_W_CHAN] NEQ 0
3     THEN  BEGIN   ! deassigning channel
3
3           STATUS=$DASSGN(CHAN=.PBCB[PBCB_W_CHAN]);
3           PBCB[PBCB_W_CHAN]=0;                ! just in case we get called
3                                               ! again after returning an error
3           IF NOT .STATUS THEN RETURN .STATUS
3
2           END;    ! deassigning channel
2
2   !+
2   ! Free the event flags now.
2   ! Ignore error if it was already free.
2   !-
2
2   IF .PBCB[PBCB_B_EFN] NEQ 0
3     THEN  BEGIN
```

```
3           STATUS=LIB$FREE_EF( %REF(.PBCB[PBCB_B_EFN]) );
4           IF (NOT .STATUS) AND (.STATUS NEQ LIB$_EF_ALRFRE)
3             THEN  RETURN .STATUS;
3           PBCB[PBCB_B_EFN]=0
2           END;
2
2   IF .PBCB[PBCB_B_ASYNC_EFN] NEQ 0
3     THEN  BEGIN
3           STATUS=LIB$FREE_EF( %REF(.PBCB[PBCB_B_ASYNC_EFN]) );
4           IF (NOT .STATUS) AND (.STATUS NEQ LIB$_EF_ALRFRE)
3             THEN  RETURN .STATUS;
3           PBCB[PBCB_B_ASYNC_EFN]=0
2           END;
2
2   !+
2   ! Free the output buffer now.
2   !-
2
2   IF .PBCB[PBCB_A_OUTPUT_BUFFER] NEQ 0
3     THEN  BEGIN   ! freeing output buffer
3
3           STATUS=LIB$FREE_VM(%REF (.PBCB[PBCB_W_OUTPUT_BUFSIZ] ),
3                                 PBCB[PBCB_A_OUTPUT_BUFFER] );
3           PBCB[PBCB_A_OUTPUT_BUFFER]=0;
3           IF NOT .STATUS THEN RETURN .STATUS
3
2           END;    ! freeing output buffer
2
2   !+
2   ! Free the output filename.
2   !-
2
2   IF .PBCB[PBCB_W_OUTNAM_LEN] NEQ 0
3     THEN  BEGIN   ! freeing outname
3           STATUS=LIB$FREE_VM(%REF (.PBCB[PBCB_W_OUTNAM_LEN] ),
3                                 PBCB[PBCB_A_OUTNAM] );
3           PBCB[PBCB_W_OUTNAM_LEN]=0;
3           IF NOT .STATUS THEN RETURN .STATUS
2           END;    ! freeing outname
2
2   !+
2   ! Close the output file, if there was one.
2   !-
2
2   IF .PBCB[PBCB_A_FAB] NEQ 0
3     THEN  BEGIN   ! Close output file
3           STATUS=$CLOSE(  FAB = .PBCB[PBCB_A_FAB]);
3           IF NOT .STATUS THEN RETURN .STATUS
2           END;    ! Close output file
2
2   !+
2   ! Free the record buffer, if there was one.
2   !-
2
2   IF .PBCB[PBCB_A_RBF] NEQ 0
3     THEN  BEGIN
3           STATUS=LIB$GET_VM(%REF(.PBCB[PBCB_W_WIDTH]+1),PBCB[PBCB_A_RBF]);
```

```
; 2211          2461 3             PBCB[PBCB_A_RBF]=0;
; 2212          2462 3             IF NOT .STATUS THEN RETURN .STATUS;
; 2213          2463 2             END;
; 2214          2464 2
; 2215          2465 2   !+
; 2216          2466 2   ! Free any FAB or RAB that was created.
; 2217          2467 2   !-
; 2218          2468 2
; 2219          2469 2   IF .PBCB[PBCB_A_FAB] NEQ 0
; 2220          2470 3     THEN  BEGIN    ! freeing FAB
; 2221          2471 3             STATUS=LIB$FREE_VM(%REF (FAB$C_BLN),
; 2222          2472 3                                PBCB[PBCB_A_FAB] );
; 2223          2473 3             PBCB[PBCB_A_FAB]=0;
; 2224          2474 3             IF NOT .STATUS THEN RETURN .STATUS
; 2225          2475 2             END;    ! freeing FAB
; 2226          2476 2
; 2227          2477 2   IF .PBCB[PBCB_A_RAB] NEQ 0
; 2228          2478 3     THEN  BEGIN    ! freeing RAB
; 2229          2479 3             STATUS=LIB$FREE_VM(%REF (RAB$C_BLN),
; 2230          2480 3                                PBCB[PBCB_A_RAB] );
; 2231          2481 3             PBCB[PBCB_A_RAB]=0;
; 2232          2482 3             IF NOT .STATUS THEN RETURN .STATUS
; 2233          2483 2             END;    ! freeing RAB
; 2234          2484 2
; 2235          2485 2   !+
; 2236          2486 2   ! Now go free the PBCB itself.
; 2237          2487 2   !-
; 2238          2488 2
; 2239          2489 3   IF NOT (STATUS=LIB$FREE_VM (%REF (PBCB_K_SIZE), PBCB))
; 2240          2490 2   THEN
; 2241          2491 2       RETURN (.STATUS);
; 2242          2492 2
; 2243          2493 2   !+
; 2244          2494 2   ! Since all went well, we can now adjust the count of how many PBCB's
; 2245          2495 2   ! we have and remove its address from the pasteboard directory.
; 2246          2496 2   !-
; 2247          2497 2
; 2248          2498 2   PBD_V_PB_AVAIL [..PBID] = 0;
; 2249          2499 2
; 2250          2500 2   PBD_L_COUNT = .PBD_L_COUNT - 1;
; 2251          2501 2
; 2252          2502 2   PBD_A_PBCB [..PBID] = 0;
; 2253          2503 2
; 2254          2504 2   RETURN  SS$_NORMAL
; 2255          2505 2
; 2256          2506 1   END;                          ! Routine SMG$DELETE_PASTEBOARD


                                                                      .EXTRN   SYS$CANEXH, SYS$DASSGN
                                                                      .EXTRN   SYS$CLOSE

                                                     0FFC 00000       .ENTRY   SMG$DELETE_PASTEBOARD, Save R2,R3,R4,R5,R6,-; 2133
                                                                               R7,R8,R9,R10,R11
                                      5B 00000000G  00   9E 00002     MOVAB    LIB$FREE_VM, R11                           ;
                                      5A 00000000'  EF   9E 00009     MOVAB    PBD_L_COUNT, R10                           ;
                                      5E            14   C2 00010     SUBL2    #20, SP                                    ;
```

```
         50            6C         01  83 00013        SUBB3   #1, (AP), DIFF                          ; 2199
                       01         50  91 00017        CMPB    DIFF, #1                                ;
                                  08  1B 0001A        BLEQU   1$                                      ;
                       50 00000000G 8F  D0 0001C        MOVL    #SMG$_WRONUMARG, R0                     ;
                                      04 00023        RET                                             ;
                       59         04  AC  D0 00024 1$:  MOVL    PBID, R9                                ; 2201
                       50         69  D0 00028        MOVL    (R9), R0                                ;
                                  0A  19 0002B        BLSS    2$                                      ;
                       6A         50  D1 0002D        CMPL    R0, PBD_L_COUNT                         ;
                                  05  14 00030        BGTR    2$                                      ;
         08        44  AA         50  E0 00032        BBS     R0, PBD_V_PB_AVAIL, 3$                  ;
                       50 00000000G 8F  D0 00037 2$:  MOVL    #SMG$_INVPAS_ID, R0                     ;
                                      04 0003E        RET                                             ;
              04  AE        04 AA40  D0 0003F 3$:  MOVL    PBD_A_PBCB[R0], PBCB                    ;
                       54         04  AE  D0 00045        MOVL    PBCB, R4                                ; 2206
                       50         54  D0 00049        MOVL    R4, R0                                  ;
                          00000000G 00  16 0004C        JSB     SMG$$BEGIN_PASTEBOARD_UPDATE_R1         ;
                       55         50  D0 00052        MOVL    R0, STATUS                              ;
                       6E         55  E9 00055        BLBC    STATUS, 7$                              ;
                       53         04  A4  D0 00058        MOVL    4(R4), CURR_PP                          ; 2213
                       54         53  D1 0005C 4$:  CMPL    CURR_PP, R4                             ; 2214
                                  1B  13 0005F        BEQL    5$                                      ;
                       52         F8  A3  9E 00061        MOVAB   -8(R3), PP_BASE                         ; 2221
                       50         10  A2  D0 00065        MOVL    16(PP_BASE), DCB                        ; 2224
                                  11  BB 00069        PUSHR   #^M<R0,R4>                              ; 2226
              0000V    CF         02  FB 0006B        CALLS   #2, SMG$$UNPASTE_VIRTUAL_DISPLAY        ;
                       55         50  D0 00070        MOVL    R0, STATUS                              ;
                       62         55  E9 00073        BLBC    STATUS, 8$                              ; 2225
                       53         0C  A2  D0 00076        MOVL    12(PP_BASE), CURR_PP                    ; 2231
                                  E0  11 0007A        BRB     4$                                      ; 2214
                          00A4    C4  D4 0007C 5$:  CLRL    164(R4)                                 ; 2234
                       02         6C  91 00080        CMPB    (AP), #2                                ; 2241
                                  13  1F 00083        BLSSU   6$                                      ;
                                  08  AC  D5 00085        TSTL    8(AP)                                   ;
                                  0E  13 00088        BEQL    6$                                      ;
                       02         6C  91 0008A        CMPB    (AP), #2                                ; 2242
                                  4C  1F 0008D        BLSSU   9$                                      ;
                                  08  AC  D5 0008F        TSTL    8(AP)                                   ;
                                  47  13 00092        BEQL    9$                                      ;
                       43         08  BC  E9 00094        BLBC    @CLEAR_SCREEN_FLAG, 9$                  ;
              0C       A4         01  8A 00098 6$:  BICB2   #1, 12(R4)                              ; 2262
                                  54  DD 0009C        PUSHL   R4                                      ; 2264
         00000000G     00         01  FB 0009E        CALLS   #1, SMG$$ERASE_PASTEBOARD               ;
                       55         50  D0 000A5        MOVL    R0, STATUS                              ;
                       2D         55  E9 000A8        BLBC    STATUS, 8$                              ;
              00E6     C4         5A  A4  B1 000AB        CMPW    90(R4), 230(R4)                         ; 2273
                                  35  13 000B1        BEQL    10$                                     ;
                       6E       00E6  C4  3C 000B3        MOVZWL  230(R4), (SP)                           ; 2276
                                4200  8F  BB 000B8        PUSHR   #^M<R9,SP>                              ; 2275
         00000000G     00         02  FB 000BC        CALLS   #2, SMG$CHANGE_PBD_CHARACTERISTICS      ;
                       55         50  D0 000C3        MOVL    R0, STATUS                              ;
                       0F         55  E9 000C6 7$:  BLBC    STATUS, 8$                              ; 2277
                                  54  DD 000C9        PUSHL   R4                                      ; 2278
         00000000G     00         01  FB 000CB        CALLS   #1, SMG$$FLUSH_BUFFER                   ;
                       55         50  D0 000D2        MOVL    R0, STATUS                              ;
                       10         55  E8 000D5        BLBS    STATUS, 10$                             ; 2279
                                023D  31 000D8 8$:  BRW     38$                                     ;
```

```
                                      54  DD 000DB 9$:    PUSHL   R4                                    ; 2285
           00000000G  00              01  FB 000DD        CALLS   #1, SMG$$FLUSH_BUFFER                 ;
                  0C  A4              01  8A 000E4        BICB2   #1, 12(R4)                            ; 2286
                      53       08     A4  D0 000E8 10$:   MOVL    8(R4), WCB                            ; 2289
                      50     00F4     C4  3C 000EC        MOVZWL  244(R4), R0                           ; 2299
                                      7A  13 000F1        BEQL    14$                                   ;
                      01              50  B1 000F3        CMPW    R0, #1                                ; 2300
                                      08  12 000F6        BNEQ    11$                                   ;
                  02  A3     00F6     C4  B1 000F8        CMPW    246(R4), 2(WCB)                       ; 2301
                                      6D  13 000FE        BEQL    14$                                   ;
                      52     0108     C4  9E 00100 11$:   MOVAB   264(R4), R2                           ; 2317
                      56     00FC     C4  9E 00105        MOVAB   252(R4), R6                           ;
                                      66  D5 0010A        TSTL    (R6)                                  ;
                                      04  12 0010C        BNEQ    12$                                   ;
                                      62  D4 0010E        CLRL    (R2)                                  ;
                                      2F  11 00110        BRB     13$                                   ;
                  08  AE              02  D0 00112 12$:   MOVL    #2, INPUT_ARGS                        ;
                  0C  AE              01  D0 00116        MOVL    #1, INPUT_ARGS+4                      ;
                  10  AE       02     A3  3C 0011A        MOVZWL  2(WCB), INPUT_ARGS+8                  ;
                               08     AE  9F 0011F        PUSHAB  INPUT_ARGS                            ;
                             0104     C4  DD 00122        PUSHL   260(R4)                               ;
                                      52  DD 00126        PUSHL   R2                                    ;
                             0100     C4  9F 00128        PUSHAB  256(R4)                               ;
                  10  AE     023C     8F  3C 0012C        MOVZWL  #572, 16(SP)                          ;
                               10     AE  9F 00132        PUSHAB  16(SP)                                ;
                                      56  DD 00135        PUSHL   R6                                    ;
           00000000G  00              06  FB 00137        CALLS   #6, SMG$GET_TERM_DATA                 ;
                      63              50  E9 0013E        BLBC    STATUS, 16$                           ;
                                      62  D5 00141 13$:   TSTL    (R2)                                  ; 2323
                                      76  13 00143        BEQL    19$                                   ;
                      58       20     A3  32 00145        CVTWL   32(WCB), FINAL_ROW                    ; 2332
                      57       22     A3  32 00149        CVTWL   34(WCB), FINAL_COL                    ; 2333
                      53     0104     C4  9E 0014D        MOVAB   260(R4), R3                           ; 2336
                                      63  DD 00152        PUSHL   (R3)                                  ;
                                      62  DD 00154        PUSHL   (R2)                                  ; 2335
                                      54  DD 00156        PUSHL   R4                                    ;
           00000000G  00              03  FB 00158        CALLS   #3, SMG$$OUTPUT                       ;
                      55              50  D0 0015F        MOVL    R0, STATUS                            ;
                      53              55  E9 00162        BLBC    STATUS, 18$                           ; 2337
                      02              6C  91 00165        CMPB    (AP), #2                              ; 2343
                                      51  1F 00168        BLSSU   19$                                   ;
                               08     AC  D5 0016A        TSTL    8(AP)                                 ;
                                      4C  13 0016D 14$:   BEQL    19$                                   ;
                      48       08     BC  E8 0016F        BLBS    @CLEAR_SCREEN_FLAG, 19$               ; 2344
                                      66  D5 00173        TSTL    (R6)                                  ; 2347
                                      04  12 00175        BNEQ    15$                                   ;
                                      62  D4 00177        CLRL    (R2)                                  ;
                                      2D  11 00179        BRB     17$                                   ;
                  08  AE              02  D0 0017B 15$:   MOVL    #2, INPUT_ARGS                        ;
                  0C  AE              58  D0 0017F        MOVL    FINAL_ROW, INPUT_ARGS+4               ;
                  10  AE              57  D0 00183        MOVL    FINAL_COL, INPUT_ARGS+8               ;
                               08     AE  9F 00187        PUSHAB  INPUT_ARGS                            ;
                                      63  DD 0018A        PUSHL   (R3)                                  ;
                                      52  DD 0018C        PUSHL   R2                                    ;
                             0100     C4  9F 0018E        PUSHAB  256(R4)                               ;
                  10  AE     023A     8F  3C 00192        MOVZWL  #570, 16(SP)                          ;
                               10     AE  9F 00198        PUSHAB  16(SP)                                ;
```

```
                                    56  DD 0019B         PUSHL   R6                                       ;
                00000000G  00       06  FB 0019D         CALLS   #6, SMG$GET_TERM_DATA                    ;
                           01       50  E8 001A4 16$:    BLBS    STATUS, 17$                              ;
                                        04 001A7         RET                                              ;
                                    63  DD 001A8 17$:    PUSHL   (R3)                                     ; 2350
                                    62  DD 001AA         PUSHL   (R2)                                     ; 2349
                                    54  DD 001AC         PUSHL   R4                                       ;
                00000000G  00       03  FB 001AE         CALLS   #3, SMG$$OUTPUT                          ;
                           55       50  D0 001B5         MOVL    R0, STATUS                               ;
                           48       55  E9 001B8 18$:    BLBC    STATUS, 22$                              ; 2351
                               74   A4  9F 001BB 19$:    PUSHAB  116(R4)                                  ; 2363
                00000000G  00       01  FB 001BE         CALLS   #1, SYS$CANEXH                           ;
                           55       50  D0 001C5         MOVL    R0, STATUS                               ;
                           09       55  E8 001C8         BLBS    STATUS, 20$                              ; 2364
                000008F8   8F       55  D1 001CB         CMPL    STATUS, #2296                            ;
                                    78  12 001D2         BNEQ    26$                                      ;
                               08   A4  D5 001D4 20$:    TSTL    8(R4)                                    ; 2371
                                    14  13 001D7         BEQL    21$                                      ;
                               08   A4  DD 001D9         PUSHL   8(R4)                                    ; 2374
                    0000V  CF       01  FB 001DC         CALLS   #1, SMG$$DEALLOCATE_WCB                  ;
                           55       50  D0 001E1         MOVL    R0, STATUS                               ;
                               08   A4  D4 001E4         CLRL    8(R4)                                    ; 2375
                           03       55  E8 001E7         BLBS    STATUS, 21$                              ; 2376
                                  0080  31 001EA         BRW     30$                                      ;
                               64   A4  B5 001ED 21$:    TSTW    100(R4)                                  ; 2387
                                    17  13 001F0         BEQL    23$                                      ;
                           7E  64   A4  3C 001F2         MOVZWL  100(R4), -(SP)                           ; 2390
                00000000G  00       01  FB 001F6         CALLS   #1, SYS$DASSGN                           ;
                           55       50  D0 001FD         MOVL    R0, STATUS                               ;
                               64   A4  B4 00200         CLRW    100(R4)                                  ; 2391
                           03       55  E8 00203 22$:    BLBS    STATUS, 23$                              ; 2393
                                  0083  31 00206         BRW     32$                                      ;
                               66   A4  95 00209 23$:    TSTB    102(R4)                                  ; 2402
                                    1F  13 0020C         BEQL    25$                                      ;
                           6E  66   A4  9A 0020E         MOVZBL  102(R4), (SP)                            ; 2404
                                    5E  DD 00212         PUSHL   SP                                       ;
                00000000G  00       01  FB 00214         CALLS   #1, LIB$FREE_EF                          ;
                           55       50  D0 0021B         MOVL    R0, STATUS                               ;
                           09       55  E8 0021E         BLBS    STATUS, 24$                              ; 2405
                00000000G  8F       55  D1 00221         CMPL    STATUS, #LIB$_EF_ALRFRE                  ;
                                    22  12 00228         BNEQ    26$                                      ;
                               66   A4  94 0022A 24$:    CLRB    102(R4)                                  ; 2407
                               67   A4  95 0022D 25$:    TSTB    103(R4)                                  ; 2410
                                    22  13 00230         BEQL    29$                                      ;
                           6E  67   A4  9A 00232         MOVZBL  103(R4), (SP)                            ; 2412
                                    5E  DD 00236         PUSHL   SP                                       ;
                00000000G  00       01  FB 00238         CALLS   #1, LIB$FREE_EF                          ;
                           55       50  D0 0023F         MOVL    R0, STATUS                               ;
                           0C       55  E8 00242         BLBS    STATUS, 28$                              ; 2413
                00000000G  8F       55  D1 00245         CMPL    STATUS, #LIB$_EF_ALRFRE                  ;
                                    03  13 0024C 26$:    BEQL    28$                                      ;
                                  00C7  31 0024E 27$:    BRW     38$                                      ;
                               67   A4  94 00251 28$:    CLRB    103(R4)                                  ; 2415
                               6C   A4  D5 00254 29$:    TSTL    108(R4)                                  ; 2422
                                    17  13 00257         BEQL    31$                                      ;
                               6C   A4  9F 00259         PUSHAB  108(R4)                                  ; 2426
                       04  AE  70   A4  3C 0025C         MOVZWL  112(R4), 4(SP)                           ; 2425
```

```
                              04   AE   9F 00261        PUSHAB  4(SP)                          ;
                     6B            02   FB 00264        CALLS   #2, LIB$FREE_VM                ; 2426
                     55            50   D0 00267        MOVL    R0, STATUS                     ;
                              6C   A4   D4 0026A        CLRL    108(R4)                        ; 2427
                     DE            55   E9 0026D 30$:   BLBC    STATUS, 27$                    ; 2428
                     50     00E4   C4   3C 00270 31$:   MOVZWL  228(R4), R0                    ; 2436
                                   18   13 00275        BEQL    33$                            ;
                            00B0   C4   9F 00277        PUSHAB  176(R4)                        ; 2439
             04      AE            50   D0 0027B        MOVL    R0, 4(SP)                      ; 2438
                              04   AE   9F 0027F        PUSHAB  4(SP)                          ;
                     6B            02   FB 00282        CALLS   #2, LIB$FREE_VM                ; 2439
                     55            50   D0 00285        MOVL    R0, STATUS                     ;
                            00E4   C4   B4 00288        CLRW    228(R4)                        ; 2440
                     BF            55   E9 0028C 32$:   BLBC    STATUS, 27$                    ; 2441
                     53     00E8   C4   9E 0028F 33$:   MOVAB   232(R4), R3                    ; 2448
                                   63   D5 00294        TSTL    (R3)                           ;
                                   0F   13 00296        BEQL    34$                            ;
                                   63   DD 00298        PUSHL   (R3)                           ; 2450
      00000000G      00            01   FB 0029A        CALLS   #1, SYS$CLOSE                  ;
                     55            50   D0 002A1        MOVL    R0, STATUS                     ;
                     71            55   E9 002A4        BLBC    STATUS, 38$                    ; 2451
                     52     00F0   C4   9E 002A7 34$:   MOVAB   240(R4), R2                    ; 2458
                                   62   D5 002AC        TSTL    (R2)                           ;
                                   1C   13 002AE        BEQL    35$                            ;
                                   52   DD 002B0        PUSHL   R2                             ; 2460
             04      AE       5A   A4   3C 002B2        MOVZWL  90(R4), 4(SP)                  ;
                              04   AE   D6 002B7        INCL    4(SP)                          ;
                              04   AE   9F 002BA        PUSHAB  4(SP)                          ;
      00000000G      00            02   FB 002BD        CALLS   #2, LIB$GET_VM                 ;
                     55            50   D0 002C4        MOVL    R0, STATUS                     ;
                                   62   D4 002C7        CLRL    (R2)                           ; 2461
                     4C            55   E9 002C9        BLBC    STATUS, 38$                    ; 2462
                                   63   D5 002CC 35$:   TSTL    (R3)                           ; 2469
                                   15   13 002CE        BEQL    36$                            ;
                                   53   DD 002D0        PUSHL   R3                             ; 2472
             04      AE       50   8F   9A 002D2        MOVZBL  #80, 4(SP)                     ; 2471
                              04   AE   9F 002D7        PUSHAB  4(SP)                          ;
                     6B            02   FB 002DA        CALLS   #2, LIB$FREE_VM                ; 2472
                     55            50   D0 002DD        MOVL    R0, STATUS                     ;
                                   63   D4 002E0        CLRL    (R3)                           ; 2473
                     33            55   E9 002E2        BLBC    STATUS, 38$                    ; 2474
                     52     00EC   C4   9E 002E5 36$:   MOVAB   236(R4), R2                    ; 2477
                                   62   D5 002EA        TSTL    (R2)                           ;
                                   15   13 002EC        BEQL    37$                            ;
                                   52   DD 002EE        PUSHL   R2                             ; 2480
             04      AE       44   8F   9A 002F0        MOVZBL  #68, 4(SP)                     ; 2479
                              04   AE   9F 002F5        PUSHAB  4(SP)                          ;
                     6B            02   FB 002F8        CALLS   #2, LIB$FREE_VM                ; 2480
                     55            50   D0 002FB        MOVL    R0, STATUS                     ;
                                   62   D4 002FE        CLRL    (R2)                           ; 2481
                     15            55   E9 00300        BLBC    STATUS, 38$                    ; 2482
                              04   AE   9F 00303 37$:   PUSHAB  PBCB                           ; 2489
             04      AE     014C   8F   3C 00306        MOVZWL  #332, 4(SP)                    ;
                              04   AE   9F 0030C        PUSHAB  4(SP)                          ;
                     6B            02   FB 0030F        CALLS   #2, LIB$FREE_VM                ;
                     55            50   D0 00312        MOVL    R0, STATUS                     ;
                     04            55   E8 00315        BLBS    STATUS, 39$                    ;
```

```
                                   50           55  D0 00318 38$:      MOVL    STATUS, R0                                        ; 2491
                                                    04 0031B           RET                                                       ;
                                   50           69  D0 0031C 39$:      MOVL    (R9), R0                                          ; 2498
                       00     44   AA           50  E5 0031F           BBCC    R0, PBD_V_PB_AVAIL, 40$                           ;
                                                6A  D7 00324 40$:      DECL    PBD_L_COUNT                                       ; 2500
                                           04 AA40  D4 00326           CLRL    PBD_A_PBCB[R0]                                    ; 2502
                                   50           01  D0 0032A           MOVL    #1, R0                                            ; 2504
                                                    04 0032D           RET                                                       ; 2506

; Routine Size:  814 bytes,    Routine Base:  _SMG$CODE + 0A95


; 2257           2507  1 !<BLF/PAGE>
```

```
 1 %SBTTL 'SMG$CREATE_VIRTUAL_DISPLAY - Create Virtual Display'
 1 GLOBAL ROUTINE SMG$CREATE_VIRTUAL_DISPLAY (
 1                                              NUM_ROWS,      ! height
 1                                              NUM_COLS,      ! width
 1                                              NEW_DISPLAY_ID,
 1                                              DISPLAY_ATTRIBUTES,
 1                                              VIDEO_ATTRIBUTES,
 1                                              CHAR_SET
 1                                             ) =
 1 !++
 1 ! FUNCTIONAL DESCRIPTION:
 1 !
 1 !       This routine creates a new virtual display -- returning its
 1 !       assigned display_id.  Its initial contents are blanks with
 1 !       video attributes set to those specified or zero.  The cursor
 1 !       will be at row 1 column 1.
 1 !
 1 ! CALLING SEQUENCE:
 1 !
 1 !       ret_status.wlc.v = SMG$CREATE_VIRTUAL_DISPLAY (
 1 !                               NUM_ROWS.rl.r,             ! Height
 1 !                               NUM_COLS.rl.r,             ! Width
 1 !                               NEW_DISPLAY_ID.wl.r
 1 !                               [,DISPLAY_ATTRIBUTES.rl.r]
 1 !                               [,VIDEO_ATTRIBUTES.rl.r]
 1 !                               [,CHAR_SET.rl.r])
 1 !
 1 ! FORMAL PARAMETERS:
 1 !
 1 !       NUM_ROWS.rl.r    Number of rows in new virtual display.
 1 !
 1 !       NUM_COLS.rl.r    Number of columns in new virtual display.
 1 !
 1 !       NEW_DISPLAY_ID.wl.r      Virtual display id of newly-created
 1 !                                virtual display.
 1 !
 1 !       DISPLAY_ATTRIBUTES.rl.r The default display attributes.
 1 !
 1 !                       SMG$M_BORDER if virtual display is to be
 1 !                                   displayed with a border.
 1 !
 1 !                       SMG$M_TRUNC_ICON if an icon should be displayed
 1 !                                   when text overflows the display bounds.
 1 !
 1 !                       SMG$M_DISPLAY_CONTROLS if carriage controls (CR, LF,
 1 !                                   FF, VT, HT) should be displayed instead
 1 !                                   of executed.
 1 !                       If omitted, none of the attributes will be set.
 1 !
 1 !       VIDEO_ATTRIBUTES.rl.r   The default rendition code to be
 1 !                       applied to all output to this display unless
 1 !                       overridden on a particular output call.
 1 !                       If not supplied, default will be all zero (no
 1 !                       attributes).
 1 !
 1 !                       Values:
 1 !
```

```
1 !                              SMG$M_BLINK     displays characters blinking.
1 !
1 !                              SMG$M_BOLD      displays characters in
1 !                                              higher-than-normal intensity.
1 !
1 !                              SMG$M_REVERSE   displays characters in reverse
1 !                                              video -- that is, using the
1 !                                              opposite default rendition of
1 !                                              the virtual display.
1 !
1 !                              SMG$M_UNDERLINE displays characters underlined.
1 !
1 !          CHAR_SET.rb.r       [Optional].  If provided, specifies the default
1 !                              character set to be used for this display.
1 !                              Recognized values are:
1 !                                              SMG$C_UNITED_KINGDOM
1 !                                              SMG$C_ASCII (default)
1 !                                              SMG$C_SPEC_GRAPHICS
1 !                                              SMG$C_ALT_CHAR
1 !                                              SMG$C_ALT_GRAPHICS
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !          NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !          NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !          SS$_NORMAL          Normal successful completion
1 !          LIB$_INSVIRMEM      Insufficient virtual memory to allocate needed
1 !                              buffer.
1 !          SMG$_INVARG         Unrecognized Video Attributes
1 !                           or Unrecognized Display Attributes
1 !          SMG$_WRONUMARG      Wrong number of arguments.
1 !
1 ! SIDE EFFECTS:
1 !
1 !          NONE
1 !--
2     BEGIN
2     BUILTIN
2          NULLPARAMETER;
2
2     $SMG$VALIDATE_ARGCOUNT (3, 6);        ! Test for right no. of args
2
3     RETURN (SMG$$CREATE_VIRTUAL_DISPLAY(
3                    .NUM_ROWS,
3                    .NUM_COLS,
3                    .NEW_DISPLAY_ID,        ! Gets the DCB address for the display created
4                    (IF NOT NULLPARAMETER(DISPLAY_ATTRIBUTES)
4                      THEN  .DISPLAY_ATTRIBUTES
3                      ELSE  UPLIT(0) ),
4                    (IF NOT NULLPARAMETER(VIDEO_ATTRIBUTES)
4                      THEN  .VIDEO_ATTRIBUTES
; 2373          2622  3                     ELSE  UPLIT(0) ),
; 2374          2623  4                   (IF NOT NULLPARAMETER(CHAR_SET)
; 2375          2624  4                     THEN  .CHAR_SET
; 2376          2625  2                     ELSE  UPLIT(0) ) ) );
; 2377          2626  2
; 2378          2627  1       END;                          ! Routine SMG$CREATE_VIRTUAL_DISPLAY



                                                     00DC3           .BLKB    1                                    :
                                           00000000  00DC4 P.AAC:   .LONG    0                                    :
                                           00000000  00DC8 P.AAD:   .LONG    0                                    :
                                           00000000  00DCC P.AAE:   .LONG    0                                    :



                                              0000 00000           .ENTRY   SMG$CREATE_VIRTUAL_DISPLAY, Save nothing      : 2509
                    50          6C          03   83 00002           SUBB3    #3, (AP), DIFF                                : 2611
                                03          50   91 00006           CMPB     DIFF, #3                                      :
                                            08   18 00009           BLEQU    1$                                            :
                                50 00000000G 8F  D0 0000B           MOVL     #SMG$_WRONUMARG, R0                          :
                                            04      00012           RET                                                    :
                                06          6C   91 00013 1$:       CMPB     (AP), #6                                      : 2623
                                            0A   1F 00016           BLSSU    2$                                            :
                                      18    AC   D5 00018           TSTL     24(AP)                                        :
                                            05   13 0001B           BEQL     2$                                            :
                                      18    AC   DD 0001D           PUSHL    CHAR_SET                                      : 2624
                                            06   11 00020           BRB      3$                                            :
                                50    D7    AF   9E 00022 2$:       MOVAB    P.AAE, R0                                     : 2625
                                            50   DD 00026           PUSHL    R0                                            :
                                05          6C   91 00028 3$:       CMPB     (AP), #5                                      : 2620
                                            0A   1F 0002B           BLSSU    4$                                            :
                                      14    AC   D5 0002D           TSTL     20(AP)                                        :
                                            05   13 00030           BEQL     4$                                            :
                                      14    AC   DD 00032           PUSHL    VIDEO_ATTRIBUTES                              : 2621
                                            06   11 00035           BRB      5$                                            :
                                50    BE    AF   9E 00037 4$:       MOVAB    P.AAD, R0                                     : 2622
                                            50   DD 0003B           PUSHL    R0                                            :
                                04          6C   91 0003D 5$:       CMPB     (AP), #4                                      : 2617
                                            0A   1F 00040           BLSSU    6$                                            :
                                      10    AC   D5 00042           TSTL     16(AP)                                        :
                                            05   13 00045           BEQL     6$                                            :
                                      10    AC   DD 00047           PUSHL    DISPLAY_ATTRIBUTES                            : 2618
                                            06   11 0004A           BRB      7$                                            :
                                50    A5    AF   9E 0004C 6$:       MOVAB    P.AAC, R0                                     : 2619
                                            50   DD 00050           PUSHL    R0                                            :
                                7E    08    AC   7D 00052 7$:       MOVQ     NUM_COLS, -(SP)                               : 2615
                                      04    AC   DD 00056           PUSHL    NUM_ROWS                                      : 2614
                          0000V CF          06   FB 00059           CALLS    #6, SMG$$CREATE_VIRTUAL_DISPLAY               :
                                                 04 0005E           RET                                                    : 2627

; Routine Size:  95 bytes,    Routine Base:  _SMG$CODE + 0DD0


; 2379          2628  1 !<BLF/PAGE>
```

```
1 %SBTTL 'SMG$DELETE_VIRTUAL_DISPLAY - Delete virtual display'
1 GLOBAL ROUTINE SMG$DELETE_VIRTUAL_DISPLAY ( DISPLAY_ID ) =
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine deletes a virtual display.  It is automatically
1 !       "unpasted" from any pasteboards on which it is pasted and
1 !       its associated buffer space is deallocated.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       ret_status.wlc.v = SMG$DELETE_VIRTUAL_DISPLAY (DISPLAY_ID.rl.r )
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       DISPLAY_ID.rl.r         Id of virtual display to be deleted.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !       SS$_NORMAL      Normal successful completion
1 !       SMG$_INVDIS_ID  Invalid display id.
1 !       SMG$_WRONUMARG  Wrong number of arguments.
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !--
2     BEGIN
2     LOCAL
2         STATUS,                         ! Status of subroutine calls
2         CURR_PP : REF $PP_DECL,         ! Addr of current pasting packet
2         DCB     : REF $DCB_DECL;        ! Addr of display control block
2
2     $SMG$VALIDATE_ARGCOUNT (1, 1);      ! Test for right no. of args
2
2     $SMG$GET_DCB ( .DISPLAY_ID, DCB);   ! Get DCB address
2
2     CURR_PP = .DCB [DCB_A_PP_NEXT];
2
2 !+
2 ! Loop through all pasteboards we're pasted to, undoing our linkage to
2 ! each.
2 !-
2
2     WHILE .CURR_PP NEQ DCB [DCB_A_PP_NEXT] ! While any remain...
2     DO
3         BEGIN   ! Overall loop
3         LOCAL
3             PBCB : REF $PBCB_DECL;      ! Addr of pasteboard control blk
```

```
3
3           PBCB = .CURR_PP [PP_A_PBCB_ADDR];
3
3           !+
3           ! Update pasting packet pointer to next pasting packet, before
3           ! the unpaste operation makes current on go away.
3           !-
3           CURR_PP = .CURR_PP [PP_A_NEXT_DCB];
3
3           !+
3           ! Now we can unpaste this linkage.
3           !-
4           IF NOT (STATUS = SMG$$UNPASTE_VIRTUAL_DISPLAY (
4                                                   .DCB,
4                                                   .PBCB ))
3           THEN
3               RETURN (.STATUS);
3
2           END;    ! Overall loop
2
2   !+
2   ! Having successfully severed our linkage with all the pasteboards to
2   ! to which we were pasted, we can now get rid of the DCB itself.
2   ! Before we can delete this DCB we must check to see if there is a
2   ! backup DCB in existance.  If so, call outselves recursively to delete
2   ! the backup DCB first.
2       IF .DCB [DCB_A_BACKUP_DCB] NEQ 0
2       THEN
3           IF NOT ( STATUS =SMG$DELETE_VIRTUAL_DISPLAY (
3                                       DCB [DCB_A_BACKUP_DCB]))
2           THEN
2               RETURN (.STATUS);
2
2   ! One remaining chore is to first release the buffer areas whose
2   ! addresses are in the DCB.  Recall that the two buffer (text and
2   ! attr) were initially allocated as a double-size buffer and split in
2   ! two.  This means we can return both at once by supplying the address
2   ! of the the text buffer and a length equal to twice its size.
2   !-
3       IF NOT (STATUS = LIB$FREE_VM ( %REF (2* .DCB [DCB_L_BUFSIZE]),
3                                   DCB [DCB_A_TEXT_BUF]))
2       THEN
2           RETURN (.STATUS);
2
2   !+
2   ! Free the line characteristics vector
2   !-
3       IF NOT (STATUS = LIB$FREE_VM (%REF ( .DCB [DCB_W_NO_ROWS] +1),
3                                   DCB [DCB_A_LINE_CHAR]))
2       THEN
2           RETURN ( .STATUS);
2
2   !+
2   ! Free the char_set buffer if there is one.
2   !-
2       IF .DCB [DCB_A_CHAR_SET_BUF] NEQ 0
2       THEN
```

```
; 2495         2743  3          IF NOT (STATUS = LIB$FREE_VM ( DCB [DCB_L_BUFSIZE],
; 2496         2744  3                                         DCB [DCB_A_CHAR_SET_BUF]))
; 2497         2745  2          THEN
; 2498         2746  2              RETURN (.STATUS);
; 2499         2747  2
; 2500         2748  2  !+
; 2501         2749  2  ! If we have a dynamic string containing a border label, free the string
; 2502         2750  2  !-
; 2503         2751  2      IF .DCB [DCB_V_BORDERED]
; 2504         2752  2      THEN
; 2505         2753  3          BEGIN   ! Bordered
; 2506         2754  3          LOCAL
; 2507         2755  3              DESC : REF BLOCK [8,BYTE];  ! Pointer to dynamic string
; 2508         2756  3                                          ! descriptor in DCB
; 2509         2757  3          DESC = DCB [DCB_Q_LABEL_DESC];
; 2510         2758  3          IF .DESC [DSC$A_POINTER] NEQ 0
; 2511         2759  3          THEN
; 2512         2760  4              IF NOT (STATUS = LIB$SFREE1_DD ( .DESC))
; 2513         2761  3              THEN
; 2514         2762  3                  RETURN (.STATUS);
; 2515         2763  2          END;    ! Bordered
; 2516         2764  2  !+
; 2517         2765  2  ! Now the DCB itself...
; 2518         2766  2  ! Before freeing this area, we clobber the byte that makes it
; 2519         2767  2  ! recognizable as a DCB.  That way, if someone inadvertantly tries to
; 2520         2768  2  ! pass us this DCB address as a DCB after having deleted the virtual
; 2521         2769  2  ! display, we can tell that it no longer is a valid DCB.
; 2522         2770  2  !-
; 2523         2771  2      DCB [DCB_B_STRUCT_TYPE] = 0;
; 2524         2772  2      RETURN (LIB$FREE_VM (%REF ( DCB_K_SIZE), DCB ));
; 2525         2773  2
; 2526         2774  1      END;                            ! Routine SMG$DELETE_VIRTUAL_DISPLAY



                                     001C 00000          .ENTRY  SMG$DELETE_VIRTUAL_DISPLAY, Save R2,R3,R4    ; 2630
            54 00000000G     00   9E 00002          MOVAB   LIB$FREE_VM, R4                              ;
            5E               08   C2 00009          SUBL2   #8, SP                                       ;
            01               6C   91 0000C          CMPB    (AP), #1                                     ; 2670
                             08   13 0000F          BEQL    1$                                           ;
            50 00000000G     8F   D0 00011          MOVL    #SMG$_WRONUMARG, R0                          ;
                                  04 00018          RET                                                  ;
            50          04   BC   D0 00019 1$:      MOVL    @DISPLAY_ID, R0                              ; 2672
      04    BC          38   A0   D1 0001D          CMPL    56(R0), @DISPLAY_ID                          ;
                             06   12 00022          BNEQ    2$                                           ;
            11          44   A0   91 00024          CMPB    68(R0), #17                                  ;
                             08   13 00028          BEQL    3$                                           ;
            50 00000000G     8F   D0 0002A 2$:      MOVL    #SMG$_INVDIS_ID, R0                          ;
                                  04 00031          RET                                                  ;
      04    AE          04   BC   D0 00032 3$:      MOVL    @DISPLAY_ID, DCB                             ;
            52          04   AE   D0 00037          MOVL    DCB, R2                                      ; 2674
            53          20   A2   D0 0003B          MOVL    32(R2), CURR_PP                              ;
            51          20   A2   9E 0003F 4$:      MOVAB   32(R2), R1                                   ; 2681
            51               53   D1 00043          CMPL    CURR_PP, R1                                  ;
                             14   13 00046          BEQL    5$                                           ;
```

```
                               51        14   A3  D0 00048        MOVL    20(CURR_PP), PBCB                            ; 2687
                               53             63  D0 0004C        MOVL    (CURR_PP), CURR_PP                           ; 2693
                                              51  DD 0004F        PUSHL   PBCB                                         ; 2700
                                              52  DD 00051        PUSHL   R2                                           ; 2699
                       0000V   CF             02  FB 00053        CALLS   #2, SMG$$UNPASTE_VIRTUAL_DISPLAY             ;
                               E4             50  E8 00058        BLBS    STATUS, 4$                                   ; 2698
                                              04     0005B        RET                                                  ; 2702
                                         40   A2  D5 0005C 5$:    TSTL    64(R2)                                       ; 2712
                                              0A  13 0005F        BEQL    6$                                           ;
                                         40   A2  9F 00061        PUSHAB  64(R2)                                       ; 2715
                               98   AF        01  FB 00064        CALLS   #1, SMG$DELETE_VIRTUAL_DISPLAY               ;
                                    61        50  E9 00068        BLBC    STATUS, 9$                                   ;
                                         10   A2  9F 0006B 6$:    PUSHAB  16(R2)                                       ; 2726
          04      AE          3C    A2        01  78 0006E        ASHL    #1, 60(R2), 4(SP)                            ; 2725
                                         04   AE  9F 00074        PUSHAB  4(SP)                                        ;
                                    64        02  FB 00077        CALLS   #2, LIB$FREE_VM                              ; 2726
                                    4F        50  E9 0007A        BLBC    STATUS, 9$                                   ;
                                         4C   A2  9F 0007D        PUSHAB  76(R2)                                       ; 2734
                              04    AE   02   A2  3C 00080        MOVZWL  2(R2), 4(SP)                                 ; 2733
                                         04   AE  D6 00085        INCL    4(SP)                                        ;
                                         04   AE  9F 00088        PUSHAB  4(SP)                                        ;
                                    64        02  FB 0008B        CALLS   #2, LIB$FREE_VM                              ; 2734
                                    3B        50  E9 0008E        BLBC    STATUS, 9$                                   ;
                                         18   A2  D5 00091        TSTL    24(R2)                                       ; 2741
                                              0C  13 00094        BEQL    7$                                           ;
                                         18   A2  9F 00096        PUSHAB  24(R2)                                       ; 2744
                                         3C   A2  9F 00099        PUSHAB  60(R2)                                       ; 2743
                                    64        02  FB 0009C        CALLS   #2, LIB$FREE_VM                              ; 2744
                                    2A        50  E9 0009F        BLBC    STATUS, 9$                                   ;
                                    15   2F   A2  E9 000A2 7$:    BLBC    47(R2), 8$                                   ; 2751
                                    51   08   A2  9E 000A6        MOVAB   8(R2), DESC                                  ; 2757
                                         04   A1  D5 000AA        TSTL    4(DESC)                                      ; 2758
                                              0C  13 000AD        BEQL    8$                                           ;
                                              51  DD 000AF        PUSHL   DESC                                         ; 2760
                         00000000G  00        01  FB 000B1        CALLS   #1, LIB$SFREE1_DD                            ;
                                    11        50  E9 000B8        BLBC    STATUS, 9$                                   ;
                                         44   A2  94 000BB 8$:    CLRB    68(R2)                                       ; 2771
                                         04   AE  9F 000BE        PUSHAB  DCB                                          ; 2772
                              04    AE   70   8F  9A 000C1        MOVZBL  #112, 4(SP)                                  ;
                                         04   AE  9F 000C6        PUSHAB  4(SP)                                        ;
                                    64        02  FB 000C9        CALLS   #2, LIB$FREE_VM                              ;
                                              04     000CC 9$:    RET                                                  ; 2774

; Routine Size:  205 bytes,    Routine Base:  _SMG$CODE + 0E2F


; 2527          2775  1 !<BLF/PAGE>
```

```
1 %SBTTL 'SMG$GET_DISPLAY_ATTR - Get  display attributes'
1 GLOBAL ROUTINE SMG$GET_DISPLAY_ATTR (
1                                 DISPLAY_ID,
1                                 HEIGHT,
1                                 WIDTH,
1                                 DISPLAY_ATTRIBUTES,
1                                 VIDEO_ATTRIBUTES,
1                                 CHAR_SET
1                                         ) =
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine returns attributes of the virtual display.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       ret_status.wlc.v = SMG$GET_DISPLAY_ATTR (
1 !                                 DISPLAY_ID.rl.r,
1 !                                 HEIGHT.wl.r,
1 !                                 WIDTH.wl.r,
1 !                                 [,DISPLAY_ATTRIBUTES.wl.r]
1 !                                 [,VIDEO_ATTRIBUTES.wl.r]
1 !                                 [,CHAR_SET.wl.r])
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       DISPLAY_ID.rl.r           The id of the display for which the
1 !                                 information is requested.
1 !
1 !       HEIGHT.wl.r               Height of display in rows
1 !
1 !       WIDTH.wl.r                Width of display in columns
1 !
1 !       DISPLAY_ATTRIBUTES.wl.r Optional.  If provided, the current
1 !                           default display attributes will be returned.
1 !                           These may be:
1 !
1 !                           SMG$M_BORDER if display is displayed with a
1 !                           border.
1 !
1 !       VIDEO_ATTRIBUTES.wl.r   Optional.   If provided, the current
1 !                           default video attributes are returned.  These
1 !                           values may be:
1 !
1 !                           SMG$M_BLINK     displays characters blinking.
1 !
1 !                           SMG$M_BOLD      displays characters in
1 !                                           higher-than-normal intensity.
1 !
1 !                           SMG$M_REVERSE   displays characters in reverse
1 !                                           video -- that is, using the
1 !                                           opposite default rendition of
1 !                                           the virtual display.
1 !
1 !                           SMG$M_UNDERLINE displays characters underlined.
1 !
1 !       CHAR_SET.wb.r   Optional.  If provided, the current default
```

```
1 !                             character set code is returned.
1 !                             Possible values are:
1 !                                             SMG$C_UNITED_KINGDOM
1 !                                             SMG$C_ASCII (default)
1 !                                             SMG$C_SPEC_GRAPHICS
1 !                                             SMG$C_ALT_CHAR
1 !                                             SMG$C_ALT_GRAPHICS
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !       SS$_NORMAL        Normal successful completion
1 !       SMG$_WRONUMARG    Wrong number of arguments
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !--
2     BEGIN
2     BUILTIN
2         NULLPARAMETER;
2
2     LOCAL
2         DCB : REF $DCB_DECL;            ! Addr of display control block
2
2     $SMG$VALIDATE_ARGCOUNT (3, 6);      ! Test for right no. of args
2
2     $SMG$GET_DCB ( .DISPLAY_ID, DCB);   ! Get DCB address
2
2     .HEIGHT = .DCB [DCB_W_NO_ROWS];
2     .WIDTH  = .DCB [DCB_W_NO_COLS];
2
2     IF NOT NULLPARAMETER (DISPLAY_ATTRIBUTES)
2     THEN .DISPLAY_ATTRIBUTES = .DCB [DCB_B_DEF_DISPLAY_ATTR];
2
2     IF NOT NULLPARAMETER (VIDEO_ATTRIBUTES)
2     THEN .VIDEO_ATTRIBUTES   = .DCB [DCB_B_DEF_VIDEO_ATTR];
2
2     IF NOT NULLPARAMETER (CHAR_SET)
2     THEN .CHAR_SET = .DCB [DCB_B_DEF_CHAR_SET];
2
2     RETURN (SS$_NORMAL);
1     END;                        ! Routine SMG$GET_DISPLAY_ATTR


                                  0000 00000        .ENTRY  SMG$GET_DISPLAY_ATTR, Save nothing         ; 2777
      50            6C         03   83 00002        SUBB3   #3, (AP), DIFF                             ; 2865
```

```
                                                                           I 1
SMG$DISPLAY_LIN SMG$DISPLAY_LINKS - Virtual Display Linkages    16-Sep-1984 00:29:22    VAX-11 Bliss-32 V4.0-742              Page  76
1-096           SMG$GET_DISPLAY_ATTR - Get  display attributes  14-Sep-1984 13:09:43    [SMGRTL.SRC]SMGDISLIN.B32;1                (15)

                        03                50  91 00006           CMPB      DIFF, #3                                   ;
                                          08  1B 00009           BLEQU     1$                                         ;
                        50 00000000G      8F  D0 0000B           MOVL      #SMG$_WRONUMARG, R0                        ;
                                              04 00012           RET                                                  ;
                        50          04    BC  D0 00013 1$:       MOVL      @DISPLAY_ID, R0                            ; 2867
                  04    BC          38    A0  D1 00017           CMPL      56(R0), @DISPLAY_ID                        ;
                                          06  12 0001C           BNEQ      2$                                         ;
                        11          44    A0  91 0001E           CMPB      68(R0), #17                                ;
                                          08  13 00022           BEQL      3$                                         ;
                        50 00000000G      8F  D0 00024 2$:       MOVL      #SMG$_INVDIS_ID, R0                        ;
                                              04 0002B           RET                                                  ;
                        50          04    BC  D0 0002C 3$:       MOVL      @DISPLAY_ID, DCB                           ;
                  08    BC          02    A0  3C 00030           MOVZWL    2(DCB), @HEIGHT                            ; 2869
                  0C    BC          06    A0  3C 00035           MOVZWL    6(DCB), @WIDTH                             ; 2870
                        04                6C  91 0003A           CMPB      (AP), #4                                   ; 2872
                                          0A  1F 0003D           BLSSU     4$                                         ;
                                    10    AC  D5 0003F           TSTL      16(AP)                                     ;
                                          05  13 00042           BEQL      4$                                         ;
                  10    BC          2F    A0  9A 00044           MOVZBL    47(DCB), @DISPLAY_ATTRIBUTES               ; 2873
                        05                6C  91 00049 4$:       CMPB      (AP), #5                                   ; 2875
                                          0A  1F 0004C           BLSSU     5$                                         ;
                                    14    AC  D5 0004E           TSTL      20(AP)                                     ;
                                          05  13 00051           BEQL      5$                                         ;
                  14    BC          2E    A0  9A 00053           MOVZBL    46(DCB), @VIDEO_ATTRIBUTES                 ; 2876
                        06                6C  91 00058 5$:       CMPB      (AP), #6                                   ; 2878
                                          0A  1F 0005B           BLSSU     6$                                         ;
                                    18    AC  D5 0005D           TSTL      24(AP)                                     ;
                                          05  13 00060           BEQL      6$                                         ;
                  18    BC          30    A0  9A 00062           MOVZBL    48(DCB), @CHAR_SET                         ; 2879
                        50                01  D0 00067 6$:       MOVL      #1, R0                                     ; 2881
                                              04 0006A           RET                                                  ; 2882

; Routine Size:  107 bytes,    Routine Base:  _SMG$CODE + 0EFC

; 2636          2883  1 !<BLF/PAGE>
```

```
%SBTTL 'SMG$LABEL_BORDER - Specify label for border'
GLOBAL ROUTINE SMG$LABEL_BORDER (
                                DISPLAY_ID,
                                LABEL_TEXT,
                                POSITION,
                                UNITS,
                                RENDITION_SET,
                                RENDITION_COMPLEMENT,
                                CHAR_SET
                              ) =
!++
! FUNCTIONAL DESCRIPTION:
!
!       This routine allows the caller to specify what label text is to
!       be used with this display.  If the specified DISPLAY_ID does not
!       have the display attribute of SMG$M_BORDER, this attribute is
!       forced.
!
!       If the text parameter is not supplied, this virtual display
!       no longer has a border text associated with it -- it becomes
!       an unlabeled border.  If a valid text string is provided, it
!       replaces the current label text for this border.
!       If the text (as positioned within the border) does not fit
!       fully within the border, SMG$_INVARG is returned.
!       E.g.
!               +-------LABEL
!               |           |
!               +-----------+
!
!
!       POSITION and UNITS as a pair specify the starting position of
!       the label text within the border.
!       If POSITION is omitted, the top border is assumed.  If UNITS is
!       omitted, a starting position will be chosen so as to center the
!       text either horizontally or vertically -- depending on implicit
!       or explicit POSITION.
!       If both are omitted, the text will be centered in the top border
!       line.
!
!       The following encoding is used:
!
!       POSITION                        UNITS                   SYMBOLIC
!       --------                        -----                   --------
!       0 = Top border line             Starting column number  SMG$K_TOP
!       1 = Bottom border line          Starting column number  SMG$K_BOTTOM
!       2 = Left border line            Starting row number     SMG$K_LEFT
!       3 = Right border line           Starting row number     SMG$K_RIGHT
!
!       Examples:
!
!       POSITION=0, UNITS=4             POSITION=1, UNITS=4
!
!       +--LABEL----------------+       +------------------------+
!       |                       |       |                        |
!       |                       |       |                        |
!       |                       |       |                        |
!       |                       |       |                        |
1 !           |                        |          |                        |
1 !           |                        |          |                        |
1 !           +------------------------+          +--LABEL-----------------+
1 !
1 !
1 !           POSITION=2, UNITS=3                 POSITION=3, UNITS=3
1 !
1 !           +------------------------+          +------------------------+
1 !           |                        |          |                        |
1 !           L                        |          |                        L
1 !           A                        |          |                        A
1 !           B                        |          |                        B
1 !           E                        |          |                        E
1 !           L                        |          |                        L
1 !           |                        |          |                        |
1 !           +------------------------+          +------------------------+
1 !
1 !
1 !   CALLING SEQUENCE:
1 !
1 !           ret_status.wlc.v = SMG$LABEL_BORDER (
1 !                                                 DISPLAY_ID.rl.r
1 !                                                 [,LABEL_TEXT.rt.dx]
1 !                                                 [,POSITION.rl.r]
1 !                                                 [,UNITS.rl.r]
1 !                                                 [,RENDITION_SET.rl.r]
1 !                                                 [,RENDITION_COMPLEMENT.rlr.]
1 !                                                 [,CHAR_SET.rl.r])
1 !
1 !   FORMAL PARAMETERS:
1 !
1 !           DISPLAY_ID.rl.r The display id of the virtual display whose
1 !                           border is to be labeled.  This display must have
1 !                           the display attribute of SMG$M_BORDER.
1 !
1 !           LABEL_TEXT.rt.dx        [Optional].  If supplied becomes the new
1 !                                   label for this display's border.  If
1 !                                   omitted, display becomes unlabeled.
1 !
1 !           POSITION.rl.r   [Optional].
1 !                           Specifies which border will contain label.
1 !                           If omitted, default to top border.
1 !
1 !           UNITS.rl.r      [Optional].
1 !                           Specifies where within the border the text label
1 !                           will start.
1 !                           If omitted, center in line indicated by POSITION
1 !
1 !           RENDITION_SET.rl.r      [Optional].
1 !                                   Each 1 bit attribute in this parameter
1 !                                   causes the corresponding attribute to
1 !                                   be set in the display.  (See below for
1 !                                   list of settable attributes.)
1 !
1 !           RENDITION_COMPLEMENT.rl.r       [Optional].
1 !                                   Each 1 bit attribute in this parameter
1 !                                   causes the corresponding attribute to
```

```
1 !                                     be complemented in the display.  (See
1 !                                     below for list of complementable
1 !                                     attributes.)
1 !
1 !         If the same bit is specified in both the RENDITION_SET parameter
1 !         and in the RENDITION_COMPLEMENT parameter, the application is
1 !         RENDITION_SET followed by RENDITION complement.  Using these two
1 !         parameters together the caller can exercise arbitrary and
1 !         independent control over each attribute on a single call.  On an
1 !         attribute by attribute basis he can cause the following
1 !         transformations:
1 !
1 !                 SET     COMPLEMENT      Action
1 !                 ---     ----------      ------
1 !                 0       0               Attribute unchanged.
1 !                 1       0               Attribute set to "on"
1 !                 0       1               Attribute set to complement of
1 !                                         current setting.
1 !                 1       1               Attribute set to "off".
1 !
1 !
1 !         Attributes which can be manipulated in this manner are:
1 !
1 !         SMG$M_BLINK  displays characters blinking.
1 !         SMG$M_BOLD  displays characters in higher-than-normal
1 !                       intensity.
1 !         SMG$M_REVERSE  displays characters in reverse video -- that is,
1 !                       using the opposite default rendition of the
1 !                       virtual display.
1 !         SMG$M_UNDERLINE  displays characters underlined.
1 !
1 !         CHAR_SET.rl.r   [Optional].  If provided, the character set to
1 !                         be used in displaying the label.
1 !                         Recognized values are:
1 !                                         SMG$C_UNITED_KINGDOM
1 !                                         SMG$C_ASCII (default)
1 !                                         SMG$C_SPEC_GRAPHICS
1 !                                         SMG$C_ALT_CHAR
1 !                                         SMG$C_ALT_GRAPHICS
1 !
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !         None
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !         None
1 !
1 ! COMPLETION STATUS:
1 !
1 !         SS$_NORMAL      Normal successful completion
1 !         SMG$_INVDIS_ID  Invalid virtual display id.
1 !         SMG$_INVARG     Positioning and/or units when considered with
1 !                         length of text results in a position that is
1 !                         outside of the border area.
1 !         SMG$_WRONUMARG  Wrong number of arguments.
```

```
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !--
2     BEGIN
2
2     LITERAL
2         K_SET_ARG = 5,
2         K_COMP_ARG= 6;
2
2     BUILTIN
2         NULLPARAMETER;
2
2     LOCAL
2         LUNITS,                             ! Implicit or explicit UNITS
2         LPOS,                               ! Implicit or explicit POSITION
2         REND_CODE,                          ! Rendition to be applied to
2                                             ! border label
2         STATUS,                             ! Status of subroutine calls
2         DESC : REF BLOCK [,BYTE],           ! Pointer to dynamic string
2                                             ! descriptor in DCB for border
2                                             ! label.
2         DCB : REF $DCB_DECL;                ! Addr. of display control block
2
2     $SMG$VALIDATE_ARGCOUNT (1, 7);          ! Test for right no. of args
2
2     $SMG$GET_DCB ( .DISPLAY_ID, DCB);       ! Get addr of DCB
2
2 !+
2 ! Get a copy of the label.
2 !-
2     DESC = DCB [DCB_Q_LABEL_DESC];
2
2     IF NULLPARAMETER (LABEL_TEXT)
2     THEN
3         BEGIN   ! No text specified
3         RETURN (LIB$SFREE1_DD ( .DESC));
2         END;    ! No text specified
2
3     IF NOT (STATUS = LIB$SCOPY_DXDX (.LABEL_TEXT, .DESC))
2     THEN
2         RETURN (.STATUS);
2
2 !+
2 ! Check to see if combination of POSITION and UNITS fit.
2 !-
3     LPOS = ( IF NOT NULLPARAMETER (POSITION) THEN ..POSITION
2                                          ELSE 0);  ! Default to top row
2
2     CASE .LPOS FROM SMG$K_TOP TO SMG$K_RIGHT OF
2     SET
2         [SMG$K_TOP,SMG$K_BOTTOM]:           ! Top or bottom row
3             BEGIN
4             LUNITS = ( IF NOT NULLPARAMETER (UNITS)
4                          THEN ..UNITS
4                          ELSE     ! Center horizontally
6                            ((.DCB[DCB_W_NO_COLS] -.DESC [DSC$W_LENGTH])
3                              / 2) + 2 );
3
3               IF .LUNITS LEQ 0                    OR
3                  .LUNITS + .DESC[DSC$W_LENGTH] GTR .DCB [DCB_W_NO_COLS] +2
3               THEN
4                   BEGIN
4                   LIB$SFREE1_DD (.DESC) ; ! Release our dynamic string
4                   RETURN (SMG$_INVARG);
3                   END;
2               END;
2
2           [SMG$K_LEFT,SMG$K_RIGHT]:           ! Left or right column
3               BEGIN
4               LUNITS = ( IF NOT NULLPARAMETER (UNITS)
4                          THEN ..UNITS
4                          ELSE     ! Center vertically
6                              ((.DCB[DCB_W_NO_ROWS] -.DESC[DSC$W_LENGTH])
3                                / 2) + 2 );
3
3               IF .LUNITS LEQ 0                    OR
3                  .LUNITS + .DESC[DSC$W_LENGTH] GTR .DCB [DCB_W_NO_ROWS] +2
3               THEN
4                   BEGIN
4                   LIB$SFREE1_DD (.DESC) ; ! Release our dynamic string
4                   RETURN (SMG$_INVARG);
3                   END;
2               END;
2
2           [OUTRANGE]:
2               RETURN (SMG$_INVARG);
2       TES;
2
2       DCB [DCB_B_LABEL_POS] = .LPOS;
2       DCB [DCB_W_LABEL_UNITS] = .LUNITS;
2
2   !+
2   ! If UNITS parameter was omitted we centered the label.  Make a note of
2   ! this fact so that if he later does a CHANGE_VIRTUAL_DISPLAY we can
2   ! again center it in its new "center".
2       DCB [DCB_V_LABEL_CENTER] = 0;
2       IF NULLPARAMETER (UNITS)
2       THEN
2           DCB [DCB_V_LABEL_CENTER] = 1;
2
2   !+
2   ! Calc. REND_CODE as a function of callers rendition arguments and
2   ! the default rendition in the DCB.
2   !-
2       $SMG$SET_REND_CODE (K_SET_ARG, K_COMP_ARG);
2                                   ! macro to use caller's args if present
2
2       DCB [DCB_B_LABEL_REND] = .REND_CODE;
2
2   !+
2   ! Deal with alternate character set.
2   !-
```

```
; 2923          3169  2     IF NOT NULLPARAMETER(CHAR_SET)
; 2924          3170  2     THEN
; 2925          3171  3         BEGIN
; 2926          3172  3         CASE ..CHAR_SET FROM SMG$C_UNITED_KINGDOM
; 2927          3173  3                         TO SMG$C_ALT_GRAPHICS OF
; 2928          3174  3         SET
; 2929          3175  3             [SMG$C_UNITED_KINGDOM,
; 2930          3176  3              SMG$C_ASCII,
; 2931          3177  3              SMG$C_SPEC_GRAPHICS,
; 2932          3178  3              SMG$C_ALT_CHAR,
; 2933          3179  3              SMG$C_ALT_GRAPHICS]:
; 2934          3180  3                             DCB [DCB_B_LABEL_CHAR_SET] = ..CHAR_SET;
; 2935          3181  3
; 2936          3182  3             [INRANGE, OUTRANGE]:
; 2937          3183  3                             RETURN (SMG$_INVARG);
; 2938          3184  3         TES;
; 2939          3185  3         END
; 2940          3186  2     ELSE        ! Use default for virtual display
; 2941          3187  2         DCB [DCB_B_LABEL_CHAR_SET] = .DCB [DCB_B_DEF_CHAR_SET];
; 2942          3188  2
; 2943          3189  2     DCB [DCB_V_BORDERED] = 1;   ! Force bordered attribute in case it
; 2944          3190  2                                 ! wasn't previously.
; 2945          3191  2 !+
; 2946          3192  2 ! We now need to recalculate the constants in the pasting packet.
; 2947          3193  2 ! We may be making the transition from unbordered to bordered, so
; 2948          3194  2 ! this virtual display now has a bigger footprint in the pasteboard
; 2949          3195  2 ! buffer, and some display which previously was not occluded may now be.
; 2950          3196  2 ! Even if we were previously bordered, the size and position of our
; 2951          3197  2 ! label may have changed.
; 2952          3198  2 ! If we are not batched at the display level, recalc. pasting packet
; 2953          3199  2 ! constants and initiate output.  Else, just remember that we need to do
; 2954          3200  2 ! it later when batch level drops to zero.
; 2955          3201  2 !-
; 2956          3202  2     IF .DCB [DCB_L_BATCH_LEVEL] EQL 0
; 2957          3203  2     THEN
; 2958          3204  3         BEGIN   ! Do it now
; 2959          3205  4         IF NOT (STATUS = SMG$$RECALC_PP_FIELDS ( .DCB))
; 2960          3206  3         THEN
; 2961          3207  3             RETURN (.STATUS);
; 2962          3208  3
; 2963          3209  3         RETURN ( SMG$$CHECK_FOR_OUTPUT_DCB ( .DCB, SMG$C_LABEL_BORDER));
; 2964          3210  3         END     ! Do it now
; 2965          3211  3
; 2966          3212  2     ELSE
; 2967          3213  2
; 2968          3214  3         BEGIN   ! Defer the action
; 2969          3215  3         DCB [DCB_V_PP_MISMATCH] =1; ! Remember for later
; 2970          3216  2         END;    ! Defer the action
; 2971          3217  2
; 2972          3218  2     RETURN (SS$_NORMAL);
; 2973          3219  1     END;                        ! Routine SMG$LABEL_BORDER




                                                00FC 00000        .ENTRY  SMG$LABEL_BORDER, Save R2,R3,R4,R5,R6,R7    ; 2885
```

```
                                57 00000000G   00   9E 00002          MOVAB   LIB$SFREE1_DD, R7             :
                      50        6C             01   83 00009          SUBB3   #1, (AP), DIFF                : 3080
                                06             50   91 0000D          CMPB    DIFF, #6                      :
                                               08   1B 00010          BLEQU   1$                            :
                                50 00000000G   8F   D0 00012          MOVL    #SMG$_WRONUMARG, R0           :
                                               04      00019          RET                                   :
                                50         04  BC   D0 0001A 1$:      MOVL    @DISPLAY_ID, R0               : 3082
                           04   BC         38  A0   D1 0001E          CMPL    56(R0), @DISPLAY_ID           :
                                               06   12 00023          BNEQ    2$                            :
                                11         44  A0   91 00025          CMPB    68(R0), #17                   :
                                               08   13 00029          BEQL    3$                            :
                                50 00000000G   8F   D0 0002B 2$:      MOVL    #SMG$_INVDIS_ID, R0           :
                                               04      00032          RET                                   :
                                52         04  BC   D0 00033 3$:      MOVL    @DISPLAY_ID, DCB              :
                                54         08  A2   9E 00037          MOVAB   8(R2), DESC                   : 3087
                                02             6C   91 0003B          CMPB    (AP), #2                      : 3089
                                               05   1F 0003E          BLSSU   4$                            :
                                           08  AC   D5 00040          TSTL    8(AP)                         :
                                               06   12 00043          BNEQ    5$                            :
                                               54   DD 00045 4$:      PUSHL   DESC                          : 3092
                                67             01   FB 00047          CALLS   #1, LIB$SFREE1_DD             :
                                               04      0004A          RET                                   :
                                               54   DD 0004B 5$:      PUSHL   DESC                          : 3095
                                           08  AC   DD 0004D          PUSHL   LABEL_TEXT                    :
                       00000000G        00     02   FB 00050          CALLS   #2, LIB$SCOPY_DXDX            :
                                56             50   D0 00057          MOVL    R0, STATUS                    :
                                03             56   E8 0005A          BLBS    STATUS, 6$                    :
                                            010B    31 0005D          BRW     29$                           :
                                03             6C   91 00060 6$:      CMPB    (AP), #3                      : 3102
                                               0B   1F 00063          BLSSU   7$                            :
                                           0C  AC   D5 00065          TSTL    12(AP)                        :
                                               06   13 00068          BEQL    7$                            :
                                55         0C  BC   D0 0006A          MOVL    @POSITION, LPOS               :
                                               02   11 0006E          BRB     8$                            :
                                               55   D4 00070 7$:      CLRL    LPOS                          :
                      03        00             55   CF 00072 8$:      CASEL   LPOS, #0, #3                  : 3105
      0039          0039      000A           000A      00076 9$:      .WORD   10$-9$,-                      :
                                                                              10$-9$,-                      :
                                                                              13$-9$,-                      :
                                                                              13$-9$                        :
                                               68   11 0007E          BRB     18$                           : 3142
                                04             6C   91 00080 10$:     CMPB    (AP), #4                      : 3109
                                               0B   1F 00083          BLSSU   11$                           :
                                           10  AC   D5 00085          TSTL    16(AP)                        :
                                               06   13 00088          BEQL    11$                           :
                                53         10  BC   D0 0008A          MOVL    @UNITS, LUNITS                : 3110
                                               11   11 0008E          BRB     12$                           :
                                50         06  A2   3C 00090 11$:     MOVZWL  6(DCB), R0                    : 3112
                                51             64   3C 00094          MOVZWL  (DESC), R1                    :
                                50             51   C2 00097          SUBL2   R1, R0                        :
                                50             02   C6 0009A          DIVL2   #2, R0                        : 3113
                                53         02  A0   9E 0009D          MOVAB   2(R0), LUNITS                 :
                                               40   15 000A1 12$:     BLEQ    17$                           : 3115
                                51             64   3C 000A3          MOVZWL  (DESC), R1                    : 3116
                                51             53   C0 000A6          ADDL2   LUNITS, R1                    :
                                50         06  A2   3C 000A9          MOVZWL  6(DCB), R0                    :
                                               2C   11 000AD          BRB     16$                           :
```

```
                         04            6C  91 000AF 13$:     CMPB    (AP), #4                                    ; 3126
                                       0B  1F 000B2          BLSSU   14$                                         ;
                                 10    AC  D5 000B4          TSTL    16(AP)                                      ;
                                       06  13 000B7          BEQL    14$                                         ;
                         53      10    BC  D0 000B9          MOVL    @UNITS, LUNITS                              ; 3127
                                       10  11 000BD          BRB     15$                                         ;
                         53      02    A2  3C 000BF 14$:     MOVZWL  2(DCB), R3                                  ; 3129
                         50            64  3C 000C3          MOVZWL  (DESC), R0                                  ;
                         53            50  C2 000C6          SUBL2   R0, R3                                      ;
                         53            02  C6 000C9          DIVL2   #2, R3                                      ; 3130
                         53            02  C0 000CC          ADDL2   #2, LUNITS                                  ;
                                       12  15 000CF 15$:     BLEQ    17$                                         ; 3132
                         51            64  3C 000D1          MOVZWL  (DESC), R1                                  ; 3133
                         51            53  C0 000D4          ADDL2   LUNITS, R1                                  ;
                         50      02    A2  3C 000D7          MOVZWL  2(DCB), R0                                  ;
                         50            02  C0 000DB 16$:     ADDL2   #2, R0                                      ;
                         50            51  D1 000DE          CMPL    R1, R0                                      ;
                                       07  15 000E1          BLEQ    19$                                         ;
                                       54  DD 000E3 17$:     PUSHL   DESC                                        ; 3136
                         67            01  FB 000E5          CALLS   #1, LIB$SFREE1_DD                           ;
                                       57  11 000E8 18$:     BRB     25$                                         ; 3137
                    31   A2            55  90 000EA 19$:     MOVB    LPOS, 49(DCB)                               ; 3145
                    2C   A2            53  B0 000EE          MOVW    LUNITS, 44(DCB)                             ; 3146
                    34   A2            04  8A 000F2          BICB2   #4, 52(DCB)                                 ; 3152
                         04            6C  91 000F6          CMPB    (AP), #4                                    ; 3153
                                       05  1F 000F9          BLSSU   20$                                         ;
                                 10    AC  D5 000FB          TSTL    16(AP)                                      ;
                                       04  12 000FE          BNEQ    21$                                         ;
                    34   A2            04  88 00100 20$:     BISB2   #4, 52(DCB)                                 ; 3155
                         50      2E    A2  9A 00104 21$:     MOVZBL  46(DCB), REND_CODE                          ; 3161
                         05            6C  91 00108          CMPB    (AP), #5                                    ;
                                       09  1F 0010B          BLSSU   22$                                         ;
                                 14    AC  D5 0010D          TSTL    20(AP)                                      ;
                                       04  13 00110          BEQL    22$                                         ;
                         50      14    BC  C8 00112          BISL2   @RENDITION_SET, REND_CODE                   ;
                         06            6C  91 00116 22$:     CMPB    (AP), #6                                    ;
                                       09  1F 00119          BLSSU   23$                                         ;
                                 18    AC  D5 0011B          TSTL    24(AP)                                      ;
                                       04  13 0011E          BEQL    23$                                         ;
                         50      18    BC  CC 00120          XORL2   @RENDITION_COMPLEMENT, REND_CODE            ;
                    33   A2            50  90 00124 23$:     MOVB    REND_CODE, 51(DCB)                          ; 3164
                         07            6C  91 00128          CMPB    (AP), #7                                    ; 3169
                                       23  1F 0012B          BLSSU   27$                                         ;
                                 1C    AC  D5 0012D          TSTL    28(AP)                                      ;
                                       1E  13 00130          BEQL    27$                                         ;
                04       00      1C    BC  CF 00132          CASEL   @CHAR_SET, #0, #4                           ; 3172
   0012         0012     0012        0012     00137 24$:     .WORD   26$-24$,-                                   ;
                                     0012     0013F                  26$-24$,-                                   ;
                                                                     26$-24$,-                                   ;
                                                                     26$-24$,-                                   ;
                                                                     26$-24$                                     ;
                         50 00000000G  8F  D0 00141 25$:     MOVL    #SMG$_INVARG, R0                            ; 3183
                                       04     00148          RET                                                 ;
                    32   A2      1C    BC  90 00149 26$:     MOVB    @CHAR_SET, 50(DCB)                          ; 3180
                                       05  11 0014E          BRB     28$                                         ; 3169
                    32   A2      30    A2  90 00150 27$:     MOVB    48(DCB), 50(DCB)                            ; 3187
                    2F   A2            01  88 00155 28$:     BISB2   #1, 47(DCB)                                 ; 3189
```

```
                                    1C    A2  D5 00159          TSTL    28(DCB)                                    ; 3202
                                          1D  12 0015C          BNEQ    31$                                        ;
                                          52  DD 0015E          PUSHL   DCB                                        ; 3205
                      0000V  CF           01  FB 00160          CALLS   #1, SMG$$RECALC_PP_FIELDS                  ;
                             56           50  D0 00165          MOVL    R0, STATUS                                 ;
                             04           56  E8 00168          BLBS    STATUS, 30$                                ;
                             50           56  D0 0016B 29$:     MOVL    STATUS, R0                                 ; 3207
                                              04 0016E          RET                                                ;
                                          1C  DD 0016F 30$:     PUSHL   #28                                        ; 3209
                                          52  DD 00171          PUSHL   DCB                                        ;
                  00000000G  00           02  FB 00173          CALLS   #2, SMG$$CHECK_FOR_OUTPUT_DCB              ;
                                              04 0017A          RET                                                ;
                         34  A2           08  88 0017B 31$:     BISB2   #8, 52(DCB)                                ; 3215
                             50           01  D0 0017F          MOVL    #1, R0                                     ; 3218
                                              04 00182          RET                                                ; 3219

; Routine Size:  387 bytes,    Routine Base:  _SMG$CODE + 0F67


; 2974          3220  1 !<BLF/PAGE>
```

```
%SBTTL 'SMG$MOVE_VIRTUAL_DISPLAY - Move previously pasted virtual display'
GLOBAL ROUTINE SMG$MOVE_VIRTUAL_DISPLAY (
                                        DISPLAY_ID,
                                        PASTEBOARD_ID,
                                        PASTEBOARD_ROW,
                                        PASTEBOARD_COL
                                      ) =
!++
! FUNCTIONAL DESCRIPTION:
!
!       The specified virtual display is moved with respect to the
!       position where it is currently pasted to the specified pasteboard
!       preserving the pasting order.  If the display is not currently
!       pasted, it is pasted at the top of the pasting order in the
!       position specified.
!       This call is not permitted while display batching is in effect.
!
! CALLING SEQUENCE:
!
!       ret_status.wlc.v = SMG$MOVE_VIRTUAL_DISPLAY (
!                                               DISPLAY_ID.rl.r,
!                                               PASTEBOARD_ID.rl.r,
!                                               PASTEBOARD_ROW.rl.r,
!                                               PASTEBOARD_COL.rl.r)
!
! FORMAL PARAMETERS:
!
!       DISPLAY_ID.rl.r         Id of virtual display to be moved.
!
!       PASTEBOARD_ID.rl.r      The pasteboard id of the pasteboard on
!                               which the movement is to take place.
!
!       PASTEBOARD_ROW.rl.r     Row on pasteboard which is to contain
!                               row 1 of the specified virtual display.
!
!       PASTEBOARD_COL.rl.r     Column on pasteboard which is to contain
!                               column 1 of the specified virtual
!                               display.
!
! IMPLICIT INPUTS:
!
!       None
!
! IMPLICIT OUTPUTS:
!
!       None
!
! COMPLETION STATUS:
!
!       SS$_NORMAL       Normal successful completion
!       SMG$_INVDIS_ID   Invalid virtual display id.
!       SMG$_INVPAS_ID   Invalid pasteboard id.
!       SMG$_WRONUMARG   Wrong number of arguments.
!       SMG$_ILLBATFNC   Display is being batched, this operation is illegal.
!
! SIDE EFFECTS:
!
```

```
1 !         NONE
1 !--
2     BEGIN
2     BUILTIN
2         AP,
2         CALLG;
2
2     LOCAL
2         STATUS,                         ! Status of subroutine calls
2
2         PP      : REF $PP_DECL,         ! Addr of the pasting packet
2         DCB     : REF $DCB_DECL,        ! Addr. of display control block
2         PBCB    : REF $PBCB_DECL;       ! Addr of pasteboard control block
2
2     $SMG$VALIDATE_ARGCOUNT (4, 4);      ! Test for right no. of args
2
2 !+
2 ! Get addresses of associated virtual display control block and
2 ! pasteboard control block, validating both the display id and the
2 ! pasteboard id.
2 !-
2     $SMG$GET_DCB ( .DISPLAY_ID, DCB);            ! Get addr of DCB
2     $SMG$GET_PBCB ( .PASTEBOARD_ID, PBCB);       ! Get addr of PBCB
2
2 !+
2 ! Give an error if the display is batched.
2 !-
2
2     IF .DCB[DCB_L_BATCH_LEVEL] NEQ 0
2     THEN
2         RETURN  SMG$_ILLBATFNC;
2
2 !+
2 ! Determine if this virtual display is already pasted to this
2 ! pasteboard.  If it is we can do the MOVE.  If it isn't we'll do a
2 ! PASTE at the specified position.
2 !-
2     IF NOT SMG$$LOCATE_PP( .DCB, .PBCB, PP)
2     THEN
2         RETURN SMG$$PASTE_VIRTUAL_DISPLAY(.DCB,.PBCB,
2                         .PASTEBOARD_ROW,.PASTEBOARD_COL);
2
2 !+
2 ! Set new row and column into pasting packet
2 !-
2     PP [PP_W_ROW]       = ..PASTEBOARD_ROW;
2     PP [PP_W_COL]       = ..PASTEBOARD_COL;
2
2 !+
2 ! Recalc. occlusions.
2 !-
3     IF NOT ( STATUS = SMG$$CHECK_OCCLUSION ( .PBCB))
2     THEN
2         RETURN (.STATUS);
2
2 !+
2 ! Recalculate the transformation constants needed to copy this display's
```

```
; 3090         3335  2 ! buffers into the associated window's buffers.
; 3091         3336  2 !-
; 3092         3337  3     IF NOT ( STATUS = SMG$$CALC_PASTE_TRANSF (.PP))
; 3093         3338  2     THEN
; 3094         3339  2         RETURN (.STATUS);
; 3095         3340  2
; 3096         3341  2     RETURN (SMG$$CHECK_FOR_OUTPUT_PBCB (.PBCB));
; 3097         3342  2
; 3098         3343  1     END;                            ! Routine SMG$MOVE_VIRTUAL_DISPLAY



                                       001C 00000          .ENTRY  SMG$MOVE_VIRTUAL_DISPLAY, Save R2,R3,R4      ; 3222
                    54 00000000'    EF   9E 00002          MOVAB   PBD_L_COUNT, R4                              ;
                    5E              04   C2 00009          SUBL2   #4, SP                                       ;
                    04              6C   91 0000C          CMPB    (AP), #4                                     ; 3292
                                    08   13 0000F          BEQL    1$                                           ;
                    50 00000000G    8F   D0 00011          MOVL    #SMG$_WRONUMARG, R0                          ;
                                    04      00018          RET                                                  ;
                    50       04     BC   D0 00019 1$:      MOVL    @DISPLAY_ID, R0                              ; 3299
               04   BC       38     A0   D1 0001D          CMPL    56(R0), @DISPLAY_ID                          ;
                                    06   12 00022          BNEQ    2$                                           ;
                    11       44     A0   91 00024          CMPB    68(R0), #17                                  ;
                                    08   13 00028          BEQL    3$                                           ;
                    50 00000000G    8F   D0 0002A 2$:      MOVL    #SMG$_INVDIS_ID, R0                          ;
                                    04      00031          RET                                                  ;
                    52       04     BC   D0 00032 3$:      MOVL    @DISPLAY_ID, DCB                             ;
                    50       08     BC   D0 00036          MOVL    @PASTEBOARD_ID, R0                           ; 3300
                                    0A   19 0003A          BLSS    4$                                           ;
                    64              50   D1 0003C          CMPL    R0, PBD_L_COUNT                              ;
                                    05   14 0003F          BGTR    4$                                           ;
     08        44   A4              50   E0 00041          BBS     R0, PBD_V_PB_AVAIL, 5$                       ;
                    50 00000000G    8F   D0 00046 4$:      MOVL    #SMG$_INVPAS_ID, R0                          ;
                                    04      0004D          RET                                                  ;
                    53       04 A440     D0 0004E 5$:      MOVL    PBD_A_PBCB[R0], PBCB                         ;
                             1C     A2   D5 00053          TSTL    28(DCB)                                      ; 3306
                                    08   13 00056          BEQL    6$                                           ;
                    50 00000000G    8F   D0 00058          MOVL    #SMG$_ILLBATFNC, R0                          ; 3308
                                    04      0005F          RET                                                  ;
                           400C     8F   BB 00060 6$:      PUSHR   #^M<R2,R3,SP>                                ; 3315
             0000V  CF              03   FB 00064          CALLS   #3, SMG$$LOCATE_PP                           ;
                    0C              50   E8 00069          BLBS    R0, 7$                                       ;
                    7E       0C     AC   7D 0006C          MOVQ    PASTEBOARD_ROW, -(SP)                        ; 3318
                                    0C   BB 00070          PUSHR   #^M<R2,R3>                                   ; 3317
             0000V  CF              04   FB 00072          CALLS   #4, SMG$$PASTE_VIRTUAL_DISPLAY               ;
                                    04      00077          RET                                                  ;
                    52              6E   D0 00078 7$:      MOVL    PP, R2                                       ; 3323
               18   A2       0C     BC   B0 0007B          MOVW    @PASTEBOARD_ROW, 24(R2)                      ;
               1A   A2       10     BC   B0 00080          MOVW    @PASTEBOARD_COL, 26(R2)                      ; 3324
                                    53   DD 00085          PUSHL   PBCB                                         ; 3329
             0000V  CF              01   FB 00087          CALLS   #1, SMG$$CHECK_OCCLUSION                     ;
                    13              50   E9 0008C          BLBC    STATUS, 8$                                   ;
                                    52   DD 0008F          PUSHL   R2                                           ; 3337
             0000V  CF              01   FB 00091          CALLS   #1, SMG$$CALC_PASTE_TRANSF                   ;
                    09              50   E9 00096          BLBC    STATUS, 8$                                   ;
```

```
                                               53  DD 00099           PUSHL   PBCB                                        ; 3341
                       00000000G  00           01  FB 0009B           CALLS   #1, SMG$$CHECK_FOR_OUTPUT_PBCB              ;
                                                   04 000A2 8$:       RET                                                 ; 3343

; Routine Size:  163 bytes,    Routine Base:  _SMG$CODE + 10EA


; 3099          3344  1 !<BLF/PAGE>
```

```
1 %SBTTL 'SMG$PASTE_VIRTUAL_DISPLAY - Paste virtual display to pasteboard'
1 GLOBAL ROUTINE SMG$PASTE_VIRTUAL_DISPLAY (
1                                               DISPLAY_ID,
1                                               PASTEBOARD_ID,
1                                               PASTEBOARD_ROW,
1                                               PASTEBOARD_COL
1                                             ) =
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       The specified virtual display is "pasted" (oriented
1 !       with respect to) a pasteboard.  This makes the display visible.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       ret_status.wlc.v = SMG$PASTE_VIRTUAL_DISPLAY (
1 !                                               DISPLAY_ID.rl.r,
1 !                                               PASTEBOARD_ID.rl.r,
1 !                                               PASTEBOARD_ROW.rl.r,
1 !                                               PASTEBOARD_COL.rl.r)
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       DISPLAY_ID.rl.r         Id of virtual display to be pasted.
1 !
1 !       PASTEBOARD_ID.rl.r      The pasteboard id of the pasteboard on
1 !                               which the pasting is to take place.
1 !
1 !       PASTEBOARD_ROW.rl.r     Row on pasteboard which is to contain
1 !                               row 1 of the specified virtual display.
1 !
1 !       PASTEBOARD_COL.rl.r     Column on pasteboard which is to contain
1 !                               column 1 of the specified virtual
1 !                               display.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       None
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       None
1 !
1 ! COMPLETION STATUS:
1 !
1 !       SS$_NORMAL      Normal successful completion
1 !       SMG$_INVDIS_ID  Invalid virtual display id.
1 !       SMG$_INVPAS_ID  Invalid pasteboard id.
1 !       SMG$_WRONUMARG  Wrong number of arguments.
1 !       SMG$_ILLBATFNC  Display is batched.
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !--
2     BEGIN
2     BUILTIN
```

```
; 3158          3402  2           AP,
; 3159          3403  2           CALLG;
; 3160          3404  2
; 3161          3405  2       LOCAL
; 3162          3406  2           STATUS,                           ! Status of subroutine calls
; 3163          3407  2
; 3164          3408  2           PP      : REF $PP_DECL,           ! Addr of the pasting packet
; 3165          3409  2                                             ! being created.
; 3166          3410  2           DCB     : REF $DCB_DECL,          ! Addr. of display control block
; 3167          3411  2           WCB     : REF $WCB_DECL,          ! Addr. of window control block
; 3168          3412  2           PBCB    : REF $PBCB_DECL;         ! Addr of pasteboard control
; 3169          3413  2                                             ! block
; 3170          3414  2
; 3171          3415  2       $SMG$VALIDATE_ARGCOUNT (4, 4);        ! Test for right no. of args
; 3172          3416  2
; 3173          3417  2   !+
; 3174          3418  2   ! Get addresses of associated virtual display control block and
; 3175          3419  2   ! pasteboard control block, validating both the display id and the
; 3176          3420  2   ! pasteboard id.
; 3177          3421  2   !-
; 3178          3422  2       $SMG$GET_DCB ( .DISPLAY_ID, DCB);    ! Get addr of DCB
; 3179          3423  2       $SMG$GET_PBCB ( .PASTEBOARD_ID, PBCB);      ! Get addr of PBCB
; 3180          3424  2
; 3181          3425  2   !+
; 3182          3426  2   ! Give an error if the display is batched.
; 3183          3427  2   !-
; 3184          3428  2
; 3185          3429  2       IF .DCB[DCB_L_BATCH_LEVEL] NEQ 0
; 3186          3430  2       THEN
; 3187          3431  2           RETURN  SMG$_ILLBATFNC;
; 3188          3432  2
; 3189          3433  2   !+
; 3190          3434  2   ! Check to make sure we're don't already have a pasting from this
; 3191          3435  2   ! virtual display to this pasteboard.  If it is, we employ the
; 3192          3436  2   ! repaste logic to remove the current pasting before allowing this new
; 3193          3437  2   ! pasting.  This is necessary because we don't want ambiguous pastings.
; 3194          3438  2   ! Note:  The repaste logic ends up recalling the paste routine
; 3195          3439  2   ! recursively (after doing an unpaste) -- but that's ok since there
; 3196          3440  2   ! can be at most one such pasting.  The second time we are called this
; 3197          3441  2   ! test will fail.
; 3198          3442  2   !-
; 3199          3443  2       IF SMG$$LOCATE_PP( .DCB, .PBCB, PP)
; 3200          3444  2       THEN
; 3201          3445  2           RETURN (CALLG (.AP, SMG$REPASTE_VIRTUAL_DISPLAY));
; 3202          3446  2
; 3203          3447  2       RETURN SMG$$PASTE_VIRTUAL_DISPLAY(.DCB,.PBCB,
; 3204          3448  2                           .PASTEBOARD_ROW,.PASTEBOARD_COL);
; 3205          3449  2
; 3206          3450  1       END;                            ! Routine SMG$PASTE_VIRTUAL_DISPLAY




                                          001C 00000        .ENTRY  SMG$PASTE_VIRTUAL_DISPLAY, Save R2,R3,R4     ; 3346
                     54 00000000'  EF  9E 00002         MOVAB   PBD_L_COUNT, R4                              ;
                     5E            04  C2 00009         SUBL2   #4, SP                                       ;
                     04           6C  91 0000C          CMPB    (AP), #4                                  : 3415
                                  08  13 0000F          BEQL    1$                                        :
                     50 00000000G 8F  D0 00011          MOVL    #SMG$_WRONUMARG, R0                       :
                                      04 00018          RET                                               :
                     50       04  BC  D0 00019 1$:      MOVL    @DISPLAY_ID, R0                           : 3422
               04    BC       38  A0  D1 0001D          CMPL    56(R0), @DISPLAY_ID                       :
                                  06  12 00022          BNEQ    2$                                        :
                     11       44  A0  91 00024          CMPB    68(R0), #17                               :
                                  08  13 00028          BEQL    3$                                        :
                     50 00000000G 8F  D0 0002A 2$:      MOVL    #SMG$_INVDIS_ID, R0                       :
                                      04 00031          RET                                               :
                     52       04  BC  D0 00032 3$:      MOVL    @DISPLAY_ID, DCB                          :
                     50       08  BC  D0 00036          MOVL    @PASTEBOARD_ID, R0                        : 3423
                                  0A  19 0003A          BLSS    4$                                        :
                     64           50  D1 0003C          CMPL    R0, PBD_L_COUNT                           :
                                  05  14 0003F          BGTR    4$                                        :
        08     44    A4           50  E0 00041          BBS     R0, PBD_V_PB_AVAIL, 5$                    :
                     50 00000000G 8F  D0 00046 4$:      MOVL    #SMG$_INVPAS_ID, R0                       :
                                      04 0004D          RET                                               :
                     53       04 A440 D0 0004E 5$:      MOVL    PBD_A_PBCB[R0], PBCB                      :
                              1C  A2  D5 00053          TSTL    28(DCB)                                   : 3429
                                  08  13 00056          BEQL    6$                                        :
                     50 00000000G 8F  D0 00058          MOVL    #SMG$_ILLBATFNC, R0                       : 3431
                                      04 0005F          RET                                               :
                            400C  8F  BB 00060 6$:      PUSHR   #^M<R2,R3,SP>                             : 3443
               0000V CF           03  FB 00064          CALLS   #3, SMG$$LOCATE_PP                        :
                     06           50  E9 00069          BLBC    R0, 7$                                    :
               0000V CF           6C  FA 0006C          CALLG   (AP), SMG$REPASTE_VIRTUAL_DISPLAY         : 3445
                                      04 00071          RET                                               :
                     7E       0C  AC  7D 00072 7$:      MOVQ    PASTEBOARD_ROW, -(SP)                     : 3448
                                  0C  BB 00076          PUSHR   #^M<R2,R3>                                : 3447
               0000V CF           04  FB 00078          CALLS   #4, SMG$$PASTE_VIRTUAL_DISPLAY            :
                                      04 0007D          RET                                               : 3450

; Routine Size:  126 bytes,    Routine Base:  _SMG$CODE + 118D


; 3207          3451  1 !<BLF/PAGE>
```

```
1 %SBTTL 'SMG$POP_VIRTUAL_DISPLAY - Pop off (delete) a sequence of virtual displays'
1 GLOBAL ROUTINE SMG$POP_VIRTUAL_DISPLAY (
1                                          DISPLAY_ID,
1                                          PASTEBOARD_ID
1                                          ) =
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This procedure deletes all the virtual displays on the specified
1 !       pasteboard, starting with the display specified, up through all
1 !       higher-pasted display.  Each of these displays is unpasted in
1 !       in the course of doing the deletion.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       ret_status.wlc.v = SMG$POP_VIRTUAL_DISPLAY ( DISPLAY_ID.rl.r,
1 !                                                    PASTEBOARD_ID.rl.r)
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       DISPLAY_ID.rl.r         Address of the display id of the lowest
1 !                               pasted virtual display to be deleted.
1 !                               All higher-pasted displays are deleted
1 !                               as well.
1 !
1 !       PASTEBOARD_ID.rl.r      Address of the pasteboard id involved.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !       SS$_NORMAL      Normal successful completion
1 !       SMG$_INVDIS_ID  Invalid display id
1 !       SMG$_INVPAS_ID  Invalid pasteboard id
1 !       SMG$_WRONUMARG  Wrong number of arguments
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !--
1
2   BEGIN
2   LOCAL
2       STATUS,                         ! Status of subr. calls
2
2       RET_STATUS,                     ! Accumulated status during
2                                       ! loop
2       PBCB : REF $PBCB_DECL,          ! Address of a pasteboard
2                                       ! control block
2
2       DCB  : REF $DCB_DECL,           ! Address of a virtual display
```

```
2                                               ! control block we started with
2
2           PP : REF $PP_DECL;                  ! Addr of 2 longwords that form
2                                               ! queue header in PP currently
2                                               ! under inspection.
2   !+
2   ! Check for right number of arguments.
2   !
2       $SMG$VALIDATE_ARGCOUNT ( 2, 2);
2
2   !+
2   ! Get addresses of virtual display control block and pasteboard control
2   ! block and validate them.
2   !-
2       $SMG$GET_PBCB ( .PASTEBOARD_ID, PBCB );
2       $SMG$GET_DCB  ( .DISPLAY_ID,    DCB);
2
2   !+
2   ! Locate the pasting packet that reflects this pasting (if one exists)
2   ! .PP is the base address of the pasting packet.
2   !-
3       IF NOT (STATUS = SMG$$LOCATE_PP ( .DCB, .PBCB, PP))
2       THEN
2           RETURN (.STATUS);
2
2   !+
2   ! Change packet address to address of queue header.
2   !-
2       PP = .PP + PP_PBCB_QUEUE_OFFSET; ! Start with specified packet
2
2       RET_STATUS = SS$_NORMAL;    ! Assume success to follow
2
2   !+
2   ! Batch the sequence of updates we are about to do.
2   !-
3       IF NOT ( STATUS = SMG$$BEGIN_PASTEBOARD_UPDATE_R1 (.PBCB))
2       THEN
2           RETURN (.STATUS);
2
2   !+
2   ! Loop for all pasting packets starting with this one to the last-pasted
2   ! one...
2   !-
2       WHILE .PP NEQ PBCB [PBCB_A_PP_NEXT]
2       DO
3           BEGIN   ! For all displays that need to be deleted
3           LOCAL
3               STATUS,                         ! Status of delete calls
3
3               PP_BASE : REF $PP_DECL,         ! Base address of the PP
3
3               DCB : REF $DCB_DECL;            ! Current virtual display that
3                                               ! needs to be deleted.
3           !+
3           ! Calc. the base address of this pasting packet since the queue
3           ! headers for this part of the chain are not at relative 0 in
3           ! the pasting packet.
```

```
; 3323          3566  3             !-
; 3324          3567  3             PP_BASE = .PP - PP_PBCB_QUEUE_OFFSET;
; 3325          3568  3
; 3326          3569  3             !+
; 3327          3570  3             ! Find DCB that is in this pairing.
; 3328          3571  3             !-
; 3329          3572  3             DCB = .PP_BASE [PP_A_DCB_ADDR];
; 3330          3573  3
; 3331          3574  3             !+
; 3332          3575  3             ! Delete this virtual display, causing it to be unpasted from
; 3333          3576  3             ! all pasteboards to which it is currently pasted.
; 3334          3577  3             !-
; 3335          3578  4             IF NOT ( STATUS = SMG$DELETE_VIRTUAL_DISPLAY ( DCB [DCB_L_DID]))
; 3336          3579  3             THEN
; 3337          3580  3                 !+
; 3338          3581  3                 ! If no error yet, save this one.
; 3339          3582  3                 !-
; 3340          3583  4                 BEGIN
; 3341          3584  4                 IF .RET_STATUS THEN RET_STATUS = .STATUS;
; 3342          3585  3                 END;
; 3343          3586  3
; 3344          3587  3             !+
; 3345          3588  3             ! Walk this chain backwards, from the packet we started with
; 3346          3589  3             ! back to the head of the chain -- since the most recently
; 3347          3590  3             ! pasted displays are at the head of the chain.
; 3348          3591  3             !-
; 3349          3592  3             PP = .PP_BASE [PP_A_PREV_PBCB];
; 3350          3593  2             END;     ! For all displays that need to be deleted
; 3351          3594  2
; 3352          3595  2
; 3353          3596  3         IF NOT (STATUS = SMG$$END_PASTEBOARD_UPDATE_R2 ( .PBCB ))
; 3354          3597  2         THEN
; 3355          3598  2             RETURN (.STATUS);
; 3356          3599  2
; 3357          3600  2         RETURN (.RET_STATUS);
; 3358          3601  2
; 3359          3602  1         END;                    ! End of routine SMG$POP_VIRTUAL_DISPLAY



                                                   007C 00000         .ENTRY  SMG$POP_VIRTUAL_DISPLAY, Save R2,R3,R4,R5,- ; 3453
                                                                              R6                                          ;
                          56 00000000'    EF   9E 00002         MOVAB   PBD_L_COUNT, R6                             ;
                          5E              04   C2 00009         SUBL2   #4, SP                                      ;
                          02              6C   91 0000C         CMPB    (AP), #2                                    ; 3517
                                          08   13 0000F         BEQL    1$                                          ;
                          50 00000000G    8F   D0 00011         MOVL    #SMG$_WRONUMARG, R0                         ;
                                               04 00018         RET                                                 ;
                          50        08    BC   D0 00019 1$:     MOVL    @PASTEBOARD_ID, R0                          ; 3523
                                          0A   19 0001D         BLSS    2$                                          ;
                          66              50   D1 0001F         CMPL    R0, PBD_L_COUNT                             ;
                                          05   14 00022         BGTR    2$                                          ;
              08      44  A6              50   E0 00024         BBS     R0, PBD_V_PB_AVAIL, 3$                      ;
                          50 00000000G    8F   D0 00029 2$:     MOVL    #SMG$_INVPAS_ID, R0                         ;
                                               04 00030         RET                                                 ;
```

```
                                55        04 A640  D0 00031 3$:     MOVL    PBD_A_PBCB[R0], PBCB                              ;
                                50        04   BC  D0 00036         MOVL    @DISPLAY_ID, R0                                   ; 3524
                        04      BC        38   A0  D1 0003A         CMPL    56(R0), @DISPLAY_ID                               ;
                                               06  12 0003F         BNEQ    4$                                                ;
                                11        44   A0  91 00041         CMPB    68(R0), #17                                       ;
                                               08  13 00045         BEQL    5$                                                ;
                                50 00000000G   8F  D0 00047 4$:     MOVL    #SMG$_INVDIS_ID, R0                               ;
                                               04  0004E            RET                                                       ;
                                50        04   BC  D0 0004F 5$:     MOVL    @DISPLAY_ID, DCB                                  ;
                                        4021   8F  BB 00053         PUSHR   #^M<R0,R5,SP>                                     ; 3530
                     0000V      CF             03  FB 00057         CALLS   #3, SMG$$LOCATE_PP                                ;
                                53             50  D0 0005C         MOVL    R0, STATUS                                        ;
                                48             53  E9 0005F         BLBC    STATUS, 9$                                        ;
                                6E             08  C0 00062         ADDL2   #8, PP                                            ; 3537
                                54             01  D0 00065         MOVL    #1, RET_STATUS                                    ; 3539
                                50             55  D0 00068         MOVL    PBCB, R0                                          ; 3544
                                   00000000G   00  16 0006B         JSB     SMG$$BEGIN_PASTEBOARD_UPDATE_R1                   ;
                                53             50  D0 00071         MOVL    R0, STATUS                                        ;
                                33             53  E9 00074         BLBC    STATUS, 9$                                        ;
                                55             6E  D1 00077 6$:     CMPL    PP, PBCB                                          ; 3552
                                               1F  13 0007A         BEQL    8$                                                ;
                52              6E             08  C3 0007C         SUBL3   #8, PP, PP_BASE                                   ; 3567
                                50        10   A2  D0 00080         MOVL    16(PP_BASE), DCB                                  ; 3572
                                          38   A0  9F 00084         PUSHAB  56(DCB)                                           ; 3578
                     FB98       CF             01  FB 00087         CALLS   #1, SMG$DELETE_VIRTUAL_DISPLAY                    ;
                                06             50  E8 0008C         BLBS    STATUS, 7$                                        ;
                                03             54  E9 0008F         BLBC    RET_STATUS, 7$                                    ; 3584
                                54             50  D0 00092         MOVL    STATUS, RET_STATUS                                ;
                                6E        0C   A2  D0 00095 7$:     MOVL    12(PP_BASE), PP                                   ; 3592
                                               DC  11 00099         BRB     6$                                                ; 3552
                                50             55  D0 0009B 8$:     MOVL    PBCB, R0                                          ; 3596
                                   00000000G   00  16 0009E         JSB     SMG$$END_PASTEBOARD_UPDATE_R2                     ;
                                53             50  D0 000A4         MOVL    R0, STATUS                                        ;
                                04             53  E8 000A7         BLBS    STATUS, 10$                                       ;
                                50             53  D0 000AA 9$:     MOVL    STATUS, R0                                        ; 3598
                                               04  000AD            RET                                                       ;
                                50             54  D0 000AE 10$:    MOVL    RET_STATUS, R0                                    ; 3600
                                               04  000B1            RET                                                       ; 3602

; Routine Size:  178 bytes,    Routine Base:  _SMG$CODE + 120B


; 3360          3603  1 !<BLF/PAGE>
```

```
1 %SBTTL 'SMG$REPASTE_VIRTUAL_DISPLAY - Repaste virtual display to pasteboard'
1 GLOBAL ROUTINE SMG$REPASTE_VIRTUAL_DISPLAY (
1                                             DISPLAY_ID,
1                                             PASTEBOARD_ID,
1                                             PASTEBOARD_ROW,
1                                             PASTEBOARD_COL
1                                           ) =
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       The specified virtual display is "unpasted" from the specified
1 !       pasteboard.  It is then "repasted" in the new position to the
1 !       same pasteboard.  The unpasting and repasting operation is done
1 !       under cover of a SMG$$BEGIN_PASTEBOARD_UPDATE_R1 and
1 !       SMG$$END_PASTEBOARD_UPDATE_R2 pair so that there is no effect on the
1 !       screen while it is going on.  Only the completed results of the
1 !       operation become visible.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       ret_status.wlc.v = SMG$REPASTE_VIRTUAL_DISPLAY (
1 !                                                       DISPLAY_ID.rl.r,
1 !                                                       PASTEBOARD_ID.rl.r,
1 !                                                       PASTEBOARD_ROW.rl.r,
1 !                                                       PASTEBOARD_COL.rl.r)
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       DISPLAY_ID.rl.r         Id of virtual display to be repasted.
1 !
1 !       PASTEBOARD_ID.rl.r      The pasteboard id of the pasteboard on
1 !                               which the unpasting/pasting is to take
1 !                               place.
1 !
1 !       PASTEBOARD_ROW.rl.r     Row on pasteboard which is to contain
1 !                               row 1 of the specified virtual display
1 !                               after repasting.
1 !
1 !       PASTEBOARD_COL.rl.r     Column on pasteboard which is to contain
1 !                               column 1 of the specified virtual
1 !                               display after repasting.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       None
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       None
1 !
1 ! COMPLETION STATUS:
1 !
1 !       SS$_NORMAL      Normal successful completion
1 !       SMG$_INVDIS_ID  Invalid virtual display id.
1 !       SMG$_INVPAS_ID  Invalid pasteboard id.
1 !       SMG$_WRONUMARG  Wrong number of arguments.
1 !
```

```
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !--
2     BEGIN
2
2     LOCAL
2         DCB     : REF $DCB_DECL,
2         PBCB    : REF $PBCB_DECL,
2         STATUS ;          ! Status of subroutine calls
2
2     $SMG$VALIDATE_ARGCOUNT (4, 4);        ! Test for right no. of args
2
2     $SMG$GET_PBCB(.PASTEBOARD_ID,PBCB);
2     $SMG$GET_DCB(.DISPLAY_ID,DCB);
2
2 !+
2 ! Set up an extra level of output inhibiting so that our UNPASTE
2 ! operation won't find its way to the screen until we're done.
2 !-
3     IF NOT (STATUS = SMG$$BEGIN_PASTEBOARD_UPDATE_R1 (.PBCB))
2     THEN
2         RETURN (.STATUS);
2
2 !+
2 ! Unpaste it from where it is.
2 !-
3     IF NOT (STATUS = SMG$$UNPASTE_VIRTUAL_DISPLAY (.DCB, .PBCB))
2     THEN
3         BEGIN
3         SMG$$END_PASTEBOARD_UPDATE_R2 (.PBCB); ! Reduce buffering level
3         RETURN (.STATUS);                       ! Return error
2         END;
2
2 !+
2 ! Now repaste to the same pasteboard in a new position.
2 !-
2     STATUS = SMG$$PASTE_VIRTUAL_DISPLAY(.DCB,.PBCB,
2                         .PASTEBOARD_ROW,.PASTEBOARD_COL);
2
2 !+
2 ! Undo one buffering level so that we are back where we started.
2 !-
2     SMG$$END_PASTEBOARD_UPDATE_R2 (.PBCB);
2
2 !+
2 ! If last PASTE operation yielded an error, return that status, else
2 ! return SS$_NORMAL;
2 !-
2     IF NOT .STATUS THEN RETURN .STATUS;
2
2     RETURN (SS$_NORMAL);
2
1     END;                          ! Routine SMG$REPASTE_VIRTUAL_DISPLAY
```

```
                                        00FC 00000          .ENTRY  SMG$REPASTE_VIRTUAL_DISPLAY, Save R2,R3,R4,-; 3605
                                                                    R5,R6,R7                                    ;
                    57 00000000G  00  9E 00002              MOVAB   SMG$$END_PASTEBOARD_UPDATE_R2, R7           ;
                    56 00000000'  EF  9E 00009              MOVAB   PBD_L_COUNT, R6                             ;
                    04            6C  91 00010              CMPB    (AP), #4                                    ; 3672
                                  08  13 00013              BEQL    1$                                          ;
                    50 00000000G  8F  D0 00015              MOVL    #SMG$_WRONUMARG, R0                         ;
                                      04 0001C              RET                                                 ;
                    50         08 BC  D0 0001D 1$:          MOVL    @PASTEBOARD_ID, R0                          ; 3674
                                  0A  19 00021              BLSS    2$                                          ;
                    66            50  D1 00023              CMPL    R0, PBD_L_COUNT                             ;
                                  05  14 00026              BGTR    2$                                          ;
     08         44  A6            50  E0 00028              BBS     R0, PBD_V_PB_AVAIL, 3$                      ;
                    50 00000000G  8F  D0 0002D 2$:          MOVL    #SMG$_INVPAS_ID, R0                         ;
                                      04 00034              RET                                                 ;
                    54         04 A640 D0 00035 3$:         MOVL    PBD_A_PBCB[R0], PBCB                        ;
                    50         04 BC  D0 0003A              MOVL    @DISPLAY_ID, R0                             ; 3675
                04  BC         38 A0  D1 0003E              CMPL    56(R0), @DISPLAY_ID                         ;
                                  06  12 00043              BNEQ    4$                                          ;
                    11         44 A0  91 00045              CMPB    68(R0), #17                                 ;
                                  08  13 00049              BEQL    5$                                          ;
                    50 00000000G  8F  D0 0004B 4$:          MOVL    #SMG$_INVDIS_ID, R0                         ;
                                      04 00052              RET                                                 ;
                    55         04 BC  D0 00053 5$:          MOVL    @DISPLAY_ID, DCB                            ;
                    50            54  D0 00057              MOVL    PBCB, R0                                    ; 3681
                       00000000G  00  16 0005A              JSB     SMG$$BEGIN_PASTEBOARD_UPDATE_R1             ;
                    53            50  D0 00060              MOVL    R0, STATUS                                  ;
                    2E            53  E9 00063              BLBC    STATUS, 7$                                  ;
                                  54  DD 00066              PUSHL   PBCB                                        ; 3688
                                  55  DD 00068              PUSHL   DCB                                         ;
             0000V  CF            02  FB 0006A              CALLS   #2, SMG$$UNPASTE_VIRTUAL_DISPLAY            ;
                    53            50  D0 0006F              MOVL    R0, STATUS                                  ;
                    07            53  E8 00072              BLBS    STATUS, 6$                                  ;
                    50            54  D0 00075              MOVL    PBCB, R0                                    ; 3691
                                  67  16 00078              JSB     SMG$$END_PASTEBOARD_UPDATE_R2               ;
                                  18  11 0007A              BRB     7$                                          ; 3692
                    7E         0C AC  7D 0007C 6$:          MOVQ    PASTEBOARD_ROW, -(SP)                       ; 3699
                                  54  DD 00080              PUSHL   PBCB                                        ; 3698
                                  55  DD 00082              PUSHL   DCB                                         ;
             0000V  CF            04  FB 00084              CALLS   #4, SMG$$PASTE_VIRTUAL_DISPLAY              ;
                    53            50  D0 00089              MOVL    R0, STATUS                                  ;
                    50            54  D0 0008C              MOVL    PBCB, R0                                    ; 3704
                                  67  16 0008F              JSB     SMG$$END_PASTEBOARD_UPDATE_R2               ;
                    04            53  E8 00091              BLBS    STATUS, 8$                                  ; 3710
                    50            53  D0 00094 7$:          MOVL    STATUS, R0                                  ;
                                      04 00097              RET                                                 ;
                    50            01  D0 00098 8$:          MOVL    #1, R0                                      ; 3712
                                      04 0009B              RET                                                 ; 3714

; Routine Size:  156 bytes,    Routine Base:  _SMG$CODE + 12BD


; 3473          3715  1 !<BLF/PAGE>
```

```
%SBTTL 'SMG$RESTORE_PHYSICAL_SCREEN - Restore physical screen'
GLOBAL ROUTINE SMG$RESTORE_PHYSICAL_SCREEN (
                                        PASTEBOARD_ID,
                                        DISPLAY_ID
                                    ) =
!++
! FUNCTIONAL DESCRIPTION:
!
!       This routine reverses the effect of SMG$SAVE_PHYSICAL_SCREEN,
!       thereby putting the physical screen back to the point it was
!       at just prior to the call to SMG$SAVE_PHYSICAL_SCREEN.
!       The display id returned by SMG$SAVE_PHYSICAL_SCREEN must be
!       passed to this routine to allow the restoration to happen.
!
! CALLING SEQUENCE:
!
!       ret_status.wlc.v = SMG$RESTORE_PHYSICAL_SCREEN (
!                                       PASTEBOARD_ID.rl.r,
!                                       DISPLAY_ID.rl.r)
!
! FORMAL PARAMETERS:
!
!       PASTEBOARD_ID.rl.r      Address of a pasteboard id which is to
!                               be "restored".
!
!       DISPLAY_ID.rl.r         Returned display id invented to
!                               perform requested function.
!                               This must be the display id returned
!                               by SMG$SAVE_FHYSCIAL_SCREEN.
!
! IMPLICIT INPUTS:
!
!       NONE
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! COMPLETION STATUS:
!
!    SS$_NORMAL          Normal successful completion
!    SMG$_INVDIS_ID      Invalid Display Id.
!    SMG$_INVPAS_ID      Invalid Pasteboard Id.
!
!
! SIDE EFFECTS:
!       NONE
!
!--

    BEGIN
    LOCAL
        DCB  : REF $DCB_DECL,   ! Address of virtual display control
                                ! block involved.

        PBCB : REF $PBCB_DECL,  ! Address of pasteboard control block

```

```
2          PP    : REF $PP_DECL,     ! Address of the pasting packet that
2                                    ! joins the virtual display to the
2                                    ! pasteboard.
2
2          STATUS;                   ! Status of subr. calls
2
2    !+
2    ! Validate number of arguments.
2    !-
2       $SMG$VALIDATE_ARGCOUNT( 2,2);
2
2    !+
2    ! Map pasteboard id into a PBCB address, and display id into a DCB addr.
2    !-
2       $SMG$GET_PBCB ( .PASTEBOARD_ID, PBCB);
2       $SMG$GET_DCB  ( .DISPLAY_ID,    DCB);
2
2    !+
2    ! Locate the pasting packet that joins this virtual display with this
2    ! pasteboard.
2    !-
3       IF NOT (STATUS = SMG$$LOCATE_PP ( .DCB, .PBCB, PP))
2       THEN
2           RETURN (.STATUS);
2
2    !+
2    ! Invalidate our knowledge of where the physical scrolling region is on
2    ! the screen, since we don't know where the non_SMG user may have left
2    ! it.
2    !-
2       PBCB [PBCB_W_TOP_SCROLL_LINE] = 0;
2       PBCB [PBCB_W_BOT_SCROLL_LINE] = 0;
2
2    !+
2    ! Determine best way to clear affected area.  If the whole screen is
2    ! involved we erase the whole screen in one operation.  If only part
2    ! of the screen is involved, we have to do it a line at at time.
2    !-
2       IF .PP [PP_W_FIRST_WCB_ROW] LEQ 1           AND
2          .PP [PP_W_LAST_WCB_ROW] GEQ .PBCB [PBCB_B_ROWS]
2       THEN
3           BEGIN   ! Full screen involved
3           !+
3           ! Clear the whole physical screen to get rid of what the non-SMG
3           ! user may have put there.
3           !-
4           IF NOT (STATUS = SMG$$ERASE_PASTEBOARD (.PBCB))
3           THEN
3               RETURN (.STATUS);
3
3           END     ! Full screen involved
3
2       ELSE
2
3           BEGIN   ! Only part of screen involved
3           !+
3           ! Clear only the part of the screen involved.  We'll have to do
```

```
3               ! it line by line.
3               ! The code to do that should really reside in module SMGMINUPD
3               ! for modularity.  However, it is here for now.
3               !-
3               LOCAL
3                   WCB : REF $WCB_DECL;        ! Addr of window control block
3                                               ! involved.
3
3               WCB = .PBCB [PBCB_A_WCB];
3
3               !+
3               ! For each line involved, set cursor to column 1 of that line
3               ! and emit erase sequence.  Setting the cursor to column 1 of
3               ! the line is necessay for non-VT100 terminals.
3               !-
3
3               INCR I FROM .PP [PP_W_FIRST_WCB_ROW] TO .PP [PP_W_LAST_WCB_ROW]
3               DO
4                   BEGIN       ! Row by row
4                   !+
4                   ! Set cursor to column 1 of row .I.
4                   !-
4                   SMG$$FIND_MIN_CURSOR_POS (
4                               .PBCB,
4                               .WCB [WCB_W_OLD_CUR_ROW],       ! Current row
4                               .WCB [WCB_W_OLD_CUR_COL],       ! Current col
4                               .I,                             ! Desired row
4                               1);                             ! Desired col
4
4                   !+
4                   ! Get escape sequence needed to erase a line.
4                   ! (Can't move this outside the loop since data is left
4                   ! in memory that FIND_MIN_CURSOR_POS might touch.
4                   !-
4
4                   $SMG$GET_TERM_DATA(ERASE_WHOLE_LINE);
4
4                   !+
4                   ! Erase the Ith line.
4                   !-
4
5                   IF NOT (STATUS = SMG$$OUTPUT ( .PBCB,
5                                   .PBCB[PBCB_L_CAP_LENGTH],
5                                   .PBCB[PBCB_A_CAP_BUFFER]))
4                   THEN
4                       RETURN (.STATUS);
4
3                   END;        ! Row by row
2               END;    ! Only part of screen involved
2
2       !+
2       ! Pop off the virtual display that SMG$SAVE_PHYSICAL_SCREEN placed on
2       ! top to cover everything up.
2       !-
3           IF NOT (STATUS = SMG$POP_VIRTUAL_DISPLAY ( .DISPLAY_ID,
3                                                      .PASTEBOARD_ID))
2           THEN
```

```
; 3646          3887  2         RETURN (.STATUS);
; 3647          3888  2
; 3648          3889  2     RETURN (SS$_NORMAL);
; 3649          3890  2
; 3650          3891  1     END;                    ! End of routine SMG$RESTORE_PHYSICAL_SCREEN



                                              03FC 00000        .ENTRY  SMG$RESTORE_PHYSICAL_SCREEN, Save R2,R3,R4,-; 3717
                                                                        R5,R6,R7,R8,R9                              :
                       59 00000000'   EF   9E 00002             MOVAB   PBD_L_COUNT, R9                             :
                       5E             14   C2 00009             SUBL2   #20, SP                                     :
                       02             6C   91 0000C             CMPB    (AP), #2                                    ; 3782
                                      08   13 0000F             BEQL    1$                                          :
                       50 00000000G   8F   D0 00011             MOVL    #SMG$_WRONUMARG, R0                         :
                                      04      00018             RET                                                 :
                       50        04   BC   D0 00019 1$:         MOVL    @PASTEBOARD_ID, R0                          ; 3787
                                      0A   19 0001D             BLSS    2$                                          :
                       69             50   D1 0001F             CMPL    R0, PBD_L_COUNT                             :
                                      05   14 00022             BGTR    2$                                          :
          08        44 A9             50   E0 00024             BBS     R0, PBD_V_PB_AVAIL, 3$                      :
                       50 00000000G   8F   D0 00029 2$:         MOVL    #SMG$_INVPAS_ID, R0                         :
                                      04      00030             RET                                                 :
                       53        04 A940   D0 00031 3$:         MOVL    PBD_A_PBCB[R0], PBCB                        :
                       50        08   BC   D0 00036             MOVL    @DISPLAY_ID, R0                             ; 3788
                    08 BC        38   A0   D1 0003A             CMPL    56(R0), @DISPLAY_ID                         :
                                      06   12 0003F             BNEQ    4$                                          :
                       11        44   A0   91 00041             CMPB    68(R0), #17                                 :
                                      08   13 00045             BEQL    5$                                          :
                       50 00000000G   8F   D0 00047 4$:         MOVL    #SMG$_INVDIS_ID, R0                         :
                                      04      0004E             RET                                                 :
                       50        08   BC   D0 0004F 5$:         MOVL    @DISPLAY_ID, DCB                            :
                                 04   AE   9F 00053             PUSHAB  PP                                          ; 3794
                                      09   BB 00056             PUSHR   #^M<R0,R3>                                  :
                 0000V CF             03   FB 00058             CALLS   #3, SMG$$LOCATE_PP                          :
                       56             50   D0 0005D             MOVL    R0, STATUS                                  :
                       27             56   E9 00060             BLBC    STATUS, 6$                                  :
                               00F4   C3   D4 00063             CLRL    244(PBCB)                                   ; 3803
                       52        04   AE   D0 00067             MOVL    PP, R2                                      ; 3811
                       01        2F   A2   B1 0006B             CMPW    47(R2), #1                                  :
                                      1C   1A 0006F             BGTRU   7$                                          :
                       50        5F   A3   9A 00071             MOVZBL  95(PBCB), R0                                ; 3812
                    31 A2             50   B1 00075             CMPW    R0, 49(R2)                                  :
                                      12   1A 00079             BGTRU   7$                                          :
                                      53   DD 0007B             PUSHL   PBCB                                        ; 3819
             00000000G 00             01   FB 0007D             CALLS   #1, SMG$$ERASE_PASTEBOARD                   :
                       56             50   D0 00084             MOVL    R0, STATUS                                  :
                       78             56   E8 00087             BLBS    STATUS, 12$                                 :
                                    0086   31 0008A 6$:         BRW     13$                                         ; 3821
                       54        08   A3   D0 0008D 7$:         MOVL    8(PBCB), WCB                                ; 3838
                       58        31   A2   3C 00091             MOVZWL  49(R2), R8                                  ; 3846
                       57      00FC   C3   9E 00095             MOVAB   252(PBCB), R7                               ; 3865
                       55      0108   C3   9E 0009A             MOVAB   264(PBCB), R5                               :
                       52        2F   A2   3C 0009F             MOVZWL  47(R2), I                                   ; 3873
                                      52   D7 000A3             DECL    I                                           :
```

```
                                    57  11 000A5         BRB     11$                                        ;
                                    01  DD 000A7 8$:     PUSHL   #1                                         ; 3852
                                    52  DD 000A9         PUSHL   I                                          ; 3856
                  7E          26    A4  32 000AB         CVTWL   38(WCB), -(SP)                             ; 3855
                  7E          24    A4  32 000AF         CVTWL   36(WCB), -(SP)                             ; 3854
                                    53  DD 000B3         PUSHL   PBCB                                       ; 3853
        00000000G 00                05  FB 000B5         CALLS   #5, SMG$$FIND_MIN_CURSOR_POS               ;
                                    67  D5 000BC         TSTL    (R7)                                       ; 3865
                                    04  12 000BE         BNEQ    9$                                         ;
                                    65  D4 000C0         CLRL    (R5)                                       ;
                                    25  11 000C2         BRB     10$                                        ;
                              08    AE  D4 000C4 9$:     CLRL    INPUT_ARGS                                 ;
                              08    AE  9F 000C7         PUSHAB  INPUT_ARGS                                 ;
                            0104    C3  DD 000CA         PUSHL   260(PBCB)                                  ;
                                    55  DD 000CE         PUSHL   R5                                         ;
                            0100    C3  9F 000D0         PUSHAB  256(PBCB)                                  ;
               10 AE        01DB    8F  3C 000D4         MOVZWL  #475, 16(SP)                               ;
                              10    AE  9F 000DA         PUSHAB  16(SP)                                     ;
                                    57  DD 000DD         PUSHL   R7                                         ;
        00000000G 00                06  FB 000DF         CALLS   #6, SMG$GET_TERM_DATA                      ;
                  31                50  E9 000E6         BLBC    STATUS, 15$                                ;
                            0104    C3  DD 000E9 10$:    PUSHL   260(PBCB)                                  ; 3873
                                    65  DD 000ED         PUSHL   (R5)                                       ; 3872
                                    53  DD 000EF         PUSHL   PBCB                                       ; 3871
        00000000G 00                03  FB 000F1         CALLS   #3, SMG$$OUTPUT                            ;
                  56                50  D0 000F8         MOVL    R0, STATUS                                 ;
                  15                56  E9 000FB         BLBC    STATUS, 13$                                ;
       A5         52                58  F3 000FE 11$:    AOBLEQ  R8, I, 8$                                  ; 3846
                              04    AC  DD 00102 12$:    PUSHL   PASTEBOARD_ID                              ; 3885
                              08    AC  DD 00105         PUSHL   DISPLAY_ID                                 ; 3884
             FDA5 CF                02  FB 00108         CALLS   #2, SMG$POP_VIRTUAL_DISPLAY                ;
                  56                50  D0 0010D         MOVL    R0, STATUS                                 ;
                  04                56  E8 00110         BLBS    STATUS, 14$                                ;
                  50                56  D0 00113 13$:    MOVL    STATUS, R0                                 ; 3887
                                        04 00116         RET                                                ;
                  50                01  D0 00117 14$:    MOVL    #1, R0                                     ; 3889
                                        04 0011A 15$:    RET                                                ; 3891

; Routine Size:  283 bytes,    Routine Base:  _SMG$CODE + 1359


; 3651          3892  1 !<BLF/PAGE>
```

```
1 %SBTTL 'SMG$SAVE_PHYSICAL_SCREEN - Save physical screen'
1 GLOBAL ROUTINE SMG$SAVE_PHYSICAL_SCREEN (
1                                              PASTEBOARD_ID,
1                                              DISPLAY_ID,
1                                              DESIRED_ROW_START,
1                                              DESIRED_ROW_END
1                                             ) =
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine should be called before calling a procedure which
1 !       may perform output to the screen without using the SMG$
1 !       This procedure saves the state of the screen so that it can be
1 !       restored via a later call to SMG$RESTORE_PHYSICAL_SCREEN.
1 !
1 !       This routine performs 4 functions:
1 !       It:
1 !           a). Creates a virtual display which is as wide as the
1 !               physical screen and is as high indicated by the
1 !               desired_row_start and desired_row_end.
1 !               The resulting virtual display id is returned
1 !               to the caller.
1 !
1 !           b). Pastes this virtual display to cover the screen at a
1 !               position corresponding to column 1 of desired_row_start.
1 !
1 !           c). Set the physical cursor to (1,1) in the virtual display.
1 !               This corresponds to (desired_row_start, 1) on the
1 !               physical screen.
1 !
1 !           d). Set the physical scrolling region to be the height
1 !               of the resulting virtual display.
1 !
1 !       If either desired_row_start or desired_row_end are omitted,
1 !       the first row of the physical display and the last row of the
1 !       physical display are used, respectively, in calculating the
1 !       height of the virtual display.
1 !
1 !       The effects of this routine can be reversed by doing a
1 !       SMG$RESTORE_PHYSICAL_SCREEN (Display_id.rl.r, Pasteboard_id.rl.r),
1 !       supplying the display_id returned by this routine.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       ret_status.wlc.v = SMG$SAVE_PHYSICAL_SCREEN (
1 !                                       PASTEBOARD_ID.rl.r,
1 !                                       DISPLAY_ID.wl.r
1 !                                       [,DESIRED_ROW_START.rl.r]
1 !                                       [,DESIRED_ROW_END.rl.r])
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       PASTEBOARD_ID.rl.r      Address of a pasteboard id which is to
1 !                               be "saved".
1 !
1 !       DISPLAY_ID.wl.r         Returned display id invented to
1 !                               perform requested function.
```

```
1 !
1 !       DESIRED_ROW_START.rl.r  Optional.  The address of the 1st row
1 !                               to be "saved".  If omitted, row 1 of
1 !                               the physical display is used.
1 !
1 !       DESIRED_ROW_END.rl.r    Optional.  The address of the last row
1 !                               to be "saved".  If omitted, the last
1 !                               row of the physical display is used.
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !               SS$_NORMAL      Normal successful completion
1 !
1 ! From: SMG$CREATE_VIRTUAL_DISPLAY
1 !               LIB$_INSVIRMEM  Insufficient virtual memory
1 !
1 ! From: SMG$PASTE_VIRTUAL_DISPLAY
1 !               SMG$_INVPAS_ID  Invalid Pasteboard Id.
1 !
1 !
1 ! SIDE EFFECTS:
1 !
1 !       The appropriate part of the physical screen will be blanked,
1 !       scrolling region will be full height of the past to be "saved",
1 !       and cursor will be at (desired_row_start,1) on screen.
1 !--
1
2   BEGIN
2   BUILTIN
2       NULLPARAMETER;
2
2   LOCAL
2       ROW1,                           ! Resulting 1st row
2       ROWN,                           ! Resulting last row
2       FULL_SCREEN,                    ! Logical indicating that we
2                                       ! are saving the whole screen.
2       PBCB : REF $PBCB_DECL,          ! Address of pasteboard control
2                                       ! block
2
2       NEW_DCB : REF $DCB_DECL,        ! Address of a display control
2                                       ! block.  This will also
2                                       ! become the display_id
2                                       ! returned.
2
2       STATUS;                         ! Status of subr. calls
2
2 !+
2 ! Validate number of arguments and get the PBCB that goes with the
2 ! Pasteboard id.
2 !-
```

```
2      $SMG$VALIDATE_ARGCOUNT( 2,4);
2
2      $SMG$GET_PBCB ( .PASTEBOARD_ID, PBCB);
2
2  !+
2  ! Assume full screen case and intialize accordingly
2  !-
2      FULL_SCREEN = 1;     ! Assume full screen
2      ROW1 = 1;
2      ROWN = .PBCB [PBCB_B_ROWS];
2
2  !+
2  ! See which optional parameters were supplied and re-adjust assumptions.
2  !-
2      IF NOT NULLPARAMETER (DESIRED_ROW_START)
2      THEN
3          BEGIN   ! Desired_row_start specified
3          FULL_SCREEN = 0;
3          ROW1 = ..DESIRED_ROW_START;
3          END;    ! Desired_row_start specified
2
2      IF NOT NULLPARAMETER (DESIRED_ROW_END)
2      THEN
3          BEGIN   ! Desired_row_end specified
3          FULL_SCREEN = 0;
3          ROWN = ..DESIRED_ROW_END;
3          END;    ! Desired_row_end specified
2  !+
2  ! If either of the optional row parameters were supplied, make sure
2  ! we got a consistant range.
2  !-
2      IF NOT .FULL_SCREEN
2      THEN
3          BEGIN   ! Validity check on rows
3          IF .ROW1 LSS 1                          OR ! Start off top
3              .ROW1 GEQ .PBCB [PBCB_B_ROWS] -1     OR ! need 2 lines to scroll
3              .ROWN LSS 1                          OR ! End off top
3              .ROWN GTR .PBCB [PBCB_B_ROWS]        OR ! End off bottom
3              .ROWN - .ROW1 LSS 1                     ! Wrong order
3          THEN
3              RETURN SMG$_INVROW;
3
2          END;    ! Validity check on rows
2
2  !+
2  ! Create a virtual display the same width as the physical screen and
2  ! as high as desired.
2  !-
3      IF NOT (STATUS = SMG$CREATE_VIRTUAL_DISPLAY (
3                          %REF ( .ROWN - .ROW1 +1),           ! # rows
3                          %REF (.PBCB [PBCB_W_WIDTH]),        ! # columns
3                          NEW_DCB,                            ! new disp. id
3                          %REF(0),                            ! default display attr
3                          %REF(0),                            ! default video attr
3                          %REF(0)                             ! default alt char set
3                          ))
2      THEN
```

```
; 3824          4064  2         RETURN (.STATUS);
; 3825          4065  2
; 3826          4066  2    !+
; 3827          4067  2    ! Paste newly-create virtual display to (desired_row_start,1) of
; 3828          4068  2    ! pasteboard.
; 3829          4069  2    !-
; 3830          4070  3     IF NOT (STATUS = SMG$$PASTE_VIRTUAL_DISPLAY (
; 3831          4071  3                         .NEW_DCB,                 ! DCB address
; 3832          4072  3                         .PBCB,                    ! Pasteboard control block
; 3833          4073  3                         ROW1,                     ! Row
; 3834          4074  3                         %REF (1)))                ! Col 1
; 3835          4075  2     THEN
; 3836          4076  2         RETURN (.STATUS);
; 3837          4077  2
; 3838          4078  2
; 3839          4079  2    !+
; 3840          4080  2    ! Set physical scrolling region to be full height of screen.
; 3841          4081  2    !-
; 3842          4082  3     IF NOT (STATUS = SMG$$FORCE_SCROLL_REG ( .PBCB,      ! Pasteboard
; 3843          4083  3                                               .ROW1,      ! Top row
; 3844          4084  3                                               .ROWN))     ! Bottom row
; 3845          4085  2     THEN
; 3846          4086  2         RETURN (.STATUS);
; 3847          4087  2
; 3848          4088  2    !+
; 3849          4089  2    ! Return id of newly-create virtual display to caller.
; 3850          4090  2    !-
; 3851          4091  2     .DISPLAY_ID = .NEW_DCB;
; 3852          4092  2
; 3853          4093  2     RETURN (SS$_NORMAL);
; 3854          4094  2
; 3855          4095  1     END;                    ! End of routine SMG$SAVE_PHYSICAL_SCREEN



                                  001C 00000        .ENTRY  SMG$SAVE_PHYSICAL_SCREEN, Save R2,R3,R4     : 3894
                   54 00000000'   EF   9E 00002     MOVAB   PBD_L_COUNT, R4                             :
                   5E             1C   C2 00009     SUBL2   #28, SP                                     :
      50           6C             02   83 0000C     SUBB3   #2, (AP), DIFF                              : 4007
                   02             50   91 00010     CMPB    DIFF, #2                                    :
                                  08   1B 00013     BLEQU   1$                                          :
                   50 00000000G   8F   D0 00015     MOVL    #SMG$_WRONUMARG, R0                         :
                                  04   0001C        RET                                                 :
                   50        04   BC   D0 0001D 1$: MOVL    @PASTEBOARD_ID, R0                          : 4009
                                  0A   19 00021     BLSS    2$                                          :
                   64             50   D1 00023     CMPL    R0, PBD_L_COUNT                             :
                                  05   14 00026     BGTR    2$                                          :
      08        44 A4             50   E0 00028     BBS     R0, PBD_V_PB_AVAIL, 3$                      :
                   50 00000000G   8F   D0 0002D 2$: MOVL    #SMG$_INVPAS_ID, R0                         :
                                  04   00034        RET                                                 :
                   53        04 A440   D0 00035 3$: MOVL    PBD_A_PBCB[R0], PBCB                        :
                   50             01   D0 0003A     MOVL    #1, FULL_SCREEN                             : 4014
                18 AE             01   D0 0003D     MOVL    #1, ROW1                                    : 4015
                   52        5F   A3   9A 00041     MOVZBL  95(PBCB), ROWN                              : 4016
                   03             6C   91 00045     CMPB    (AP), #3                                    : 4021
```

```
                                                                      C  4
SMG$DISPLAY_LIN SMG$DISPLAY_LINKS - Virtual Display Linkages     16-Sep-1984 00:29:22    VAX-11 Bliss-32 V4.0-742          Page 109
1-096           SMG$SAVE_PHYSICAL_SCREEN - Save physical screen 14-Sep-1984 13:09:43    [SMGRTL.SRC]SMGDISLIN.B32;1          (22)

                                                            0C  1F 00048          BLSSU   4$                                            ;
                                                       0C   AC  D5 0004A          TSTL    12(AP)                                        ;
                                                            07  13 0004D          BEQL    4$                                            ;
                                                            50  D4 0004F          CLRL    FULL_SCREEN                                   ; 4024
                                         18   AE       0C   BC  D0 00051          MOVL    @DESIRED_ROW_START, ROW1                      ; 4025
                                              04            6C  91 00056 4$:      CMPB    (AP), #4                                      ; 4028
                                                            0B  1F 00059          BLSSU   5$                                            ;
                                                       10   AC  D5 0005B          TSTL    16(AP)                                        ;
                                                            06  13 0005E          BEQL    5$                                            ;
                                                            50  D4 00060          CLRL    FULL_SCREEN                                   ; 4031
                                              52       10   BC  D0 00062          MOVL    @DESIRED_ROW_END, ROWN                        ; 4032
                                              2F            50  E8 00066 5$:      BLBS    FULL_SCREEN, 7$                               ; 4038
                                                       18   AE  D5 00069          TSTL    ROW1                                          ; 4041
                                                            22  15 0006C          BLEQ    6$                                            ;
                                              50       5F   A3  9A 0006E          MOVZBL  95(PBCB), R0                                  ; 4042
                                                            50  D7 00072          DECL    R0                                            ;
                                              50       18   AE  D1 00074          CMPL    ROW1, R0                                      ;
                                                            16  18 00078          BGEQ    6$                                            ;
                                                            52  D5 0007A          TSTL    ROWN                                          ; 4043
                                                            12  15 0007C          BLEQ    6$                                            ;
                52       5F    A3             08            00  ED 0007E          CMPZV   #0, #8, 95(PBCB), ROWN                        ; 4044
                                                            0A  19 00084          BLSS    6$                                            ;
                               50        18   AE            01  C1 00086          ADDL3   #1, ROW1, R0                                  ; 4045
                                              50            52  D1 0008B          CMPL    ROWN, R0                                      ;
                                                            08  18 0008E          BGEQ    7$                                            ;
                                              50 00000000G  8F  D0 00090 6$:      MOVL    #SMG$_INVROW, R0                              ; 4047
                                                                04 00097          RET                                                   ;
                                                       10   AE  D4 00098 7$:      CLRL    16(SP)                                        ; 4061
                                                       10   AE  9F 0009B          PUSHAB  16(SP)                                        ;
                                                       10   AE  D4 0009E          CLRL    16(SP)                                        ; 4060
                                                       10   AE  9F 000A1          PUSHAB  16(SP)                                        ;
                                                       10   AE  D4 000A4          CLRL    16(SP)                                        ; 4059
                                                       10   AE  9F 000A7          PUSHAB  16(SP)                                        ;
                                                       20   AE  9F 000AA          PUSHAB  NEW_DCB                                       ; 4055
                                         14   AE       5A   A3  3C 000AD          MOVZWL  90(PBCB), 20(SP)                              ; 4057
                                                       14   AE  9F 000B2          PUSHAB  20(SP)                                        ;
                               50             52       2C   AE  C3 000B5          SUBL3   ROW1, ROWN, R0                                ; 4056
                                         14   AE       01   A0  9E 000BA          MOVAB   1(R0), 20(SP)                                 ;
                                                       14   AE  9F 000BF          PUSHAB  20(SP)                                        ;
                                      0000V   CF            06  FB 000C2          CALLS   #6, SMG$$CREATE_VIRTUAL_DISPLAY               ;
                                              30            50  E9 000C7          BLBC    STATUS, 8$                                    ; 4055
                                         10   AE            01  D0 000CA          MOVL    #1, 16(SP)                                    ; 4074
                                                       10   AE  9F 000CE          PUSHAB  16(SP)                                        ;
                                                       1C   AE  9F 000D1          PUSHAB  ROW1                                          ; 4070
                                                            53  DD 000D4          PUSHL   PBCB                                          ; 4072
                                                       20   AE  DD 000D6          PUSHL   NEW_DCB                                       ; 4071
                                      0000V   CF            04  FB 000D9          CALLS   #4, SMG$$PASTE_VIRTUAL_DISPLAY                ;
                                              19            50  E9 000DE          BLBC    STATUS, 8$                                    ; 4070
                                                            52  DD 000E1          PUSHL   ROWN                                          ; 4084
                                                       1C   AE  DD 000E3          PUSHL   ROW1                                          ; 4083
                                                            53  DD 000E6          PUSHL   PBCB                                          ; 4082
                                  00000000G   00            03  FB 000E8          CALLS   #3, SMG$$FORCE_SCROLL_REG                     ;
                                              08            50  E9 000EF          BLBC    STATUS, 8$                                    ;
                                         08   BC       14   AE  D0 000F2          MOVL    NEW_DCB, @DISPLAY_ID                          ; 4091
                                              50            01  D0 000F7          MOVL    #1, R0                                        ; 4093
                                                                04 000FA 8$:      RET                                                   ; 4095
```

```
; Routine Size:  251 bytes,    Routine Base:  _SMG$CODE + 1474

; 3856          4096  1 !<BLF/PAGE>
```

```
1 %SBTTL 'SMG$SET_DISPLAY_SCROLL_REGION - Set scrolling region in a virtual display'
1 GLOBAL ROUTINE SMG$SET_DISPLAY_SCROLL_REGION (
1                                          DISPLAY_ID,
1                                          TOP_LINE_OF_REGION,
1                                          BOTTOM_LINE_OF_REGION
1                                         ) =
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine sets the top and bottom lines of a 'scrolling region'
1 !       in a virtual display.  The scrolling region limits are used by
1 !       output routines which scroll (SMG$PUT_WITH_SCROLL and
1 !       SMG$PUT_LINE with line advancing).  If this routine is called
1 !       with only a display_id, the scrolling region defaults to the
1 !       entire display.
1 !
1 !       If a top and bottom line are passed, they must be within the
1 !       display bounds.  Scrolling can not occur outside the bounds of
1 !       a display.
1 !
1 !       This routine does not change the appearance of the screen or the
1 !       cursor position.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       ret_status.wlc.v = SMG$SET_DISPLAY_SCROLL_REGION (
1 !                                                  DISPLAY_ID.rl.r
1 !                                                  [,TOP_LINE_OF_REGION.rl.r]
1 !                                                  [,BOTTOM_LINE_OF_REGION.rl.r])
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       DISPLAY_ID.rl.r         Display id of desired display.
1 !
1 !       TOP_LINE_OF_REGION.rl.r Optional.  The top line of a scrolling
1 !                               region.  Defaults to line 1 of the display.
1 !
1 !       BOTTOM_LINE_OF_REGION.rl.r
1 !                               Optional.  The bottom line of a scrolling
1 !                               region.  Defaults to the bottom line of the
1 !                               display.
1 !
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !       SS$_NORMAL      Normal successful completion
1 !       SMG$_INVARG     Bottom line is less than or equal to top line.
1 !       SMG$_INVROW     Row number is negative or too large
1 !       SMG$_WRONUMARG  Wrong number arguments.
```

```
!
! SIDE EFFECTS:
!
!       NONE
!--
    BEGIN
    BUILTIN
        NULLPARAMETER;

    LOCAL
        TOP_LINE,                       ! working top line
        BOTTOM_LINE,                    ! working bottom line
        DCB : REF $DCB_DECL;            ! Addr. of display control block

    $SMG$VALIDATE_ARGCOUNT (1,3);

    $SMG$GET_DCB ( .DISPLAY_ID, DCB);   ! Get address of display control
                                        ! block

!+
! Validate optional arguments.
!-

    TOP_LINE = .DCB [DCB_W_ROW_START];  ! init to default

    IF NOT NULLPARAMETER (TOP_LINE_OF_REGION)
    THEN
        BEGIN
        IF ..TOP_LINE_OF_REGION GEQ .DCB [DCB_W_ROW_START] AND
            ..TOP_LINE_OF_REGION LEQ .DCB [DCB_W_NO_ROWS]
        THEN
            TOP_LINE = ..TOP_LINE_OF_REGION
        ELSE
            RETURN (SMG$_INVROW);       ! can't be outside display
        END;

    BOTTOM_LINE = .DCB [DCB_W_NO_ROWS]; ! init to default

    IF NOT NULLPARAMETER (BOTTOM_LINE_OF_REGION)
    THEN
        BEGIN
        IF ..BOTTOM_LINE_OF_REGION GEQ .DCB [DCB_W_ROW_START] AND
            ..BOTTOM_LINE_OF_REGION LEQ .DCB [DCB_W_NO_ROWS]
        THEN
            BOTTOM_LINE = ..BOTTOM_LINE_OF_REGION
        ELSE
            RETURN (SMG$_INVROW);       ! can't be outside display
        END;

    IF .BOTTOM_LINE LEQ .TOP_LINE
    THEN
        RETURN (SMG$_INVARG);           ! can't go backwards or
                                        ! overlap

!+
! If we get here, we have a valid scrolling region.  Store it.
!-
```

```
; 3972          4211  2
; 3973          4212  2      DCB [DCB_W_TOP_OF_SCRREG] = .TOP_LINE;
; 3974          4213  2      DCB [DCB_W_BOTTOM_OF_SCRREG] = .BOTTOM_LINE;
; 3975          4214  2
; 3976          4215  2      RETURN (SS$_NORMAL);
; 3977          4216  1      END;                    ! end of routine SMG$SET_DISPLAY_SCROLL_REGION





                                                  0004 00000          .ENTRY   SMG$SET_DISPLAY_SCROLL_REGION, Save R2           ; 4098
                    50          6C           01  83 00002          SUBB3    #1, (AP), DIFF                                    ; 4168
                                02           50  91 00006          CMPB     DIFF, #2                                          :
                                             08  1B 00009          BLEQU    1$                                                :
                                50 00000000G 8F  D0 0000B          MOVL     #SMG$_WRONUMARG, R0                               :
                                             04     00012          RET                                                        :
                                50        04 BC  D0 00013 1$:      MOVL     @DISPLAY_ID, R0                                   ; 4170
                          04    BC        38 A0  D1 00017          CMPL     56(R0), @DISPLAY_ID                               :
                                             06  12 0001C          BNEQ     2$                                                :
                                11        44 A0  91 0001E          CMPB     68(R0), #17                                       :
                                             08  13 00022          BEQL     3$                                                :
                                50 00000000G 8F  D0 00024 2$:      MOVL     #SMG$_INVDIS_ID, R0                               :
                                             04     0002B          RET                                                        :
                                50        04 BC  D0 0002C 3$:      MOVL     @DISPLAY_ID, DCB                                  :
                                52           60  3C 00030          MOVZWL   (DCB), TOP_LINE                                   ; 4177
                                02           6C  91 00033          CMPB     (AP), #2                                          ; 4179
                                             1A  1F 00036          BLSSU    4$                                                :
                                          08 AC  D5 00038          TSTL     8(AP)                                             :
                                             15  13 0003B          BEQL     4$                                                :
    08   BC           60        10           00  ED 0003D          CMPZV    #0, #16, (DCB), @TOP_LINE_OF_REGION               ; 4182
                                             32  14 00043          BGTR     5$                                                :
    08   BC      02   A0        10           00  ED 00045          CMPZV    #0, #16, 2(DCB), @TOP_LINE_OF_REGION              ; 4183
                                             29  19 0004C          BLSS     5$                                                :
                                52        08 BC  D0 0004E          MOVL     @TOP_LINE_OF_REGION, TOP_LINE                     ; 4185
                                51        02 A0  3C 00052 4$:      MOVZWL   2(DCB), BOTTOM_LINE                               ; 4190
                                03           6C  91 00056          CMPB     (AP), #3                                          ; 4192
                                             24  1F 00059          BLSSU    6$                                                :
                                          0C AC  D5 0005B          TSTL     12(AP)                                            :
                                             1F  13 0005E          BEQL     6$                                                :
    0C   BC           60        10           00  ED 00060          CMPZV    #0, #16, (DCB), @BOTTOM_LINE_OF_REGION            ; 4195
                                             0F  14 00066          BGTR     5$                                                :
    0C   BC      02   A0        10           00  ED 00068          CMPZV    #0, #16, 2(DCB), @BOTTOM_LINE_OF_REGION           ; 4196
                                             06  19 0006F          BLSS     5$                                                :
                                51        0C BC  D0 00071          MOVL     @BOTTOM_LINE_OF_REGION, BOTTOM_LINE               ; 4198
                                             08  11 00075          BRB      6$                                                :
                                50 00000000G 8F  D0 00077 5$:      MOVL     #SMG$_INVROW, R0                                  ; 4200
                                             04     0007E          RET                                                        :
                                52           51  D1 0007F 6$:      CMPL     BOTTOM_LINE, TOP_LINE                             ; 4203
                                             08  14 00082          BGTR     7$                                                :
                                50 00000000G 8F  D0 00084          MOVL     #SMG$_INVARG, R0                                  ; 4205
                                             04     0008B          RET                                                        :
                          48    A0           52  B0 0008C 7$:      MOVW     TOP_LINE, 72(DCB)                                 ; 4212
                          4A    A0           51  B0 00090          MOVW     BOTTOM_LINE, 74(DCB)                              ; 4213
                                50           01  D0 00094          MOVL     #1, R0                                            ; 4215
                                             04     00097          RET                                                        ; 4216
```

```
; Routine Size:  152 bytes,    Routine Base:  _SMG$CODE + 156F

; 3978          4217  1 !<BLF/PAGE>
```

```
1 %SBTTL 'SMG$UNPASTE_VIRTUAL_DISPLAY - Unpaste virtual display from pasteboard'
1 GLOBAL ROUTINE SMG$UNPASTE_VIRTUAL_DISPLAY (
1                                               DISPLAY_ID,
1                                               PASTEBOARD_ID
1                                               ) =
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       The specified virtual display is "unpasted" from a pasteboard.
1 !       Unpasting does not destroy the virtual display or its contents.
1 !       It merely removes its mapping to a particular pasteboard and
1 !       hence its visibility on that pasteboard.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       ret_status.wlc.v = SMG$UNPASTE_VIRTUAL_DISPLAY (
1 !                                               DISPLAY_ID.rl.r,
1 !                                               PASTEBOARD_ID.rl.r)
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       DISPLAY_ID.rl.r         Id of virtual display to be unpasted.
1 !
1 !       PASTEBOARD_ID.rl.r      The pasteboard id of the pasteboard from
1 !                               which the unpasting is to take place.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       None
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       None
1 !
1 ! COMPLETION STATUS:
1 !
1 !       SS$_NORMAL        Normal successful completion
1 !       SMG$_INVDIS_ID    Invalid virtual display id.
1 !       SMG$_INVPAS_ID    Invalid pasteboard id.
1 !       SMG$_WRONUMARG    Wrong number of arguments.
1 !       SMG$_NOTPASTED    Specified virtual display is not currently
1 !                         pasted to the specified pasteboard.
1 !       SMG$_ILLBATFNC    Display is batched.
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !--
2     BEGIN
2     LOCAL
2       STATUS,                         ! Status of subroutine call
2
2       DCB     : REF $DCB_DECL,        ! Addr of display control block
2       PBCB    : REF $PBCB_DECL;       ! Addr of pasteboard control block
2
2     $SMG$VALIDATE_ARGCOUNT (2, 2);    ! Test for right no. of args
2
```

```
                                                                        J  4
SMG$DISPLAY_LIN SMG$DISPLAY_LINKS - Virtual Display Linkages     16-Sep-1984 00:29:22    VAX-11 Bliss-32 V4.0-742             Page 116
1-096           SMG$UNPASTE_VIRTUAL_DISPLAY - Unpaste virtual d 14-Sep-1984 13:09:43    [SMGRTL.SRC]SMGDISLIN.B32;1               (24)

; 4037          4275  2 !+
; 4038          4276  2 ! Get addresses of virtual display control block and pasteboard control
; 4039          4277  2 ! block and validate display id and pasteboard id.
; 4040          4278  2 !-
; 4041          4279  2     $SMG$GET_PBCB ( .PASTEBOARD_ID, PBCB);
; 4042          4280  2                                           ! Get addr of pasteboard control
; 4043          4281  2                                           ! block
; 4044          4282  2
; 4045          4283  2     $SMG$GET_DCB ( .DISPLAY_ID, DCB);     ! Get addr of virtual display
; 4046          4284  2                                           ! control block
; 4047          4285  2
; 4048          4286  2 !+
; 4049          4287  2 ! Give an error if the display is batched.
; 4050          4288  2 !-
; 4051          4289  2
; 4052          4290  2     IF .DCB[DCB_L_BATCH_LEVEL] NEQ 0
; 4053          4291  2     THEN
; 4054          4292  2         RETURN  SMG$_ILLBATFNC;
; 4055          4293  2
; 4056          4294  2     RETURN SMG$$UNPASTE_VIRTUAL_DISPLAY(.DCB,.PBCB)
; 4057          4295  2
; 4058          4296  1     END;                                  ! Routine SMG$UNPASTE_VIRTUAL_DISPLAY


                                 0004 00000          .ENTRY  SMG$UNPASTE_VIRTUAL_DISPLAY, Save R2                 ; 4219
                   52 00000000'  EF   9E 00002          MOVAB   PBD_L_COUNT, R2
                   02            6C   91 00009          CMPB    (AP), #2                                             ; 4273
                                 08   13 0000C          BEQL    1$
                   50 00000000G  8F   D0 0000E          MOVL    #SMG$_WRONUMARG, R0
                                      04 00015          RET
                   50        08  BC   D0 00016 1$:      MOVL    @PASTEBOARD_ID, R0                                   ; 4279
                                 0A   19 0001A          BLSS    2$
                   62            50   D1 0001C          CMPL    R0, PBD_L_COUNT
                                 05   14 0001F          BGTR    2$
         08        44   A2       50   E0 00021          BBS     R0, PBD_V_PB_AVAIL, 3$
                   50 00000000G  8F   D0 00026 2$:      MOVL    #SMG$_INVPAS_ID, R0
                                      04 0002D          RET
                   51        04 A240  D0 0002E 3$:      MOVL    PBD_A_PBCB[R0], PBCB
                   50        04  BC   D0 00033          MOVL    @DISPLAY_ID, R0                                      ; 4283
              04   BC        38  A0   D1 00037          CMPL    56(R0), @DISPLAY_ID
                                 06   12 0003C          BNEQ    4$
                   11        44  A0   91 0003E          CMPB    68(R0), #17
                                 08   13 00042          BEQL    5$
                   50 00000000G  8F   D0 00044 4$:      MOVL    #SMG$_INVDIS_ID, R0
                                      04 0004B          RET
                   50        04  BC   D0 0004C 5$:      MOVL    @DISPLAY_ID, DCB
                             1C  A0   D5 00050          TSTL    28(DCB)                                              ; 4290
                                 08   13 00053          BEQL    6$
                   50 00000000G  8F   D0 00055          MOVL    #SMG$_ILLBATFNC, R0                                  ; 4292
                                      04 0005C          RET
                                 03   BB 0005D 6$:      PUSHR   #^M<R0,R1>                                           ; 4294
            0000V  CF            02   FB 0005F          CALLS   #2, SMG$$UNPASTE_VIRTUAL_DISPLAY
                                      04 00064          RET                                                          ; 4296
```

```
; Routine Size:  101 bytes,    Routine Base:  _SMG$CODE + 1607

; 4059          4297  1 !<BLF/PAGE>
```

```
1 %SBTTL 'SMG$$CALC_PASTE_TRANSF - Calculate pasting transformation constants'
1 GLOBAL ROUTINE SMG$$CALC_PASTE_TRANSF ( PP  : REF $PP_DECL ) =
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This procedure precalculates the constants needed to efficiently
1 !       copy portions of the text and attributes from the virtual
1 !       display buffers located in the DCB to the window buffer located
1 !       in the WCB.
1 !       This data is derived from the pasting relationship between
1 !       the given virtual display and the pasteboard to which it is
1 !       pasted.  The calculated constants are stored in the pasting
1 !       packet that reflects this pasting.
1 !
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       ret_status.wlc.v = SMG$$CALC_PASTE_TRANSF ( PP.mab.r)
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       PP.mab.r        Address of pasting packet.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !       SS$_NORMAL      Normal successful completion
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !--
1
2   BEGIN
2   LOCAL
2       TEMP    : BLOCK [8,BYTE],       ! Temporary representation of
2                                       ! display buffer area as
2                                       ! projected on window buffer.
2       DCB     : REF $DCB_DECL,        ! Addr of DCB involved.
2       WCB     : REF $WCB_DECL,        ! Addr of WCB involved.
2       PBCB    : REF $PBCB_DECL,       ! Addr of PBCB involved.
2       OVERLAP : BLOCK [8,BYTE];       ! Describes area of overlap
2                                       ! between virtual display and
2                                       ! window buffer.
2
2   PBCB = .PP [PP_A_PBCB_ADDR] ;
2   WCB = .PBCB [PBCB_A_WCB] ;
2   DCB = .PP [PP_A_DCB_ADDR] ;
2
2 !+
2 ! Mark the border label as being invisible until it proves otherwise.
2 !-
2       PP [PP_W_LABEL_BYTES_TO_MOVE] = 0;
2       PP [PP_W_SRC_LABEL_OFF]       = 0;
2       PP [PP_W_DST_LABEL_OFF]       = 0;
2
2
2       TEMP [DCB_W_ROW_START] = .DCB [DCB_W_ROW_START] + .PP [PP_W_ROW]-1;
2       TEMP [DCB_W_NO_ROWS]   = .DCB [DCB_W_NO_ROWS];
2       TEMP [DCB_W_COL_START] = .DCB [DCB_W_COL_START] + .PP [PP_W_COL]-1;
2       TEMP [DCB_W_NO_COLS]   = .DCB [DCB_W_NO_COLS];
2
2 !+
2 ! Check to see what part (if any) of this virtual display maps onto
2 ! the viewable part of the pasteboard -- i.e., the area that goes into
2 ! the window control block buffer.
2 !-
2       IF NOT SMG$$OCCLUDE (
2                               WCB [WCB_Q_COORD],      ! Area of window buffer
2                               TEMP,                   ! Area of display buffer
2                               OVERLAP )               ! Area of overlap
2                                                       ! (if any)
2       THEN
3           BEGIN   ! No overlap
3           PP [PP_W_ROWS_TO_MOVE] = 0 ; ! There are no rows to move
3           !+
3           ! If the display isn't visible, the border label isn't visible
3           ! either.    **** Not really true -- clean this up later ****
3           !-
3           END     ! No overlap
2       ELSE
3           BEGIN   ! Overlap
3           LOCAL
3               DCB_START_ROW,          ! 1st row of display buffer that lands
3                                       ! in window buffer.
3               DCB_START_COL;          ! 1st column of display buffer that
3                                       ! lands in window buffer.
3
3           PP [PP_W_ROWS_TO_MOVE] = .OVERLAP [DCB_W_NO_ROWS];
3           PP [PP_W_MOVE_LENGTH]  = .OVERLAP [DCB_W_NO_COLS];
3
3           PP [PP_W_FIRST_WCB_ROW] = .OVERLAP [DCB_W_ROW_START];
3           PP [PP_W_LAST_WCB_ROW]  = .OVERLAP [DCB_W_ROW_START] +
3                                     .OVERLAP [DCB_W_NO_ROWS] - 1;
3
3           PP [PP_W_FIRST_WCB_COL] = .OVERLAP [DCB_W_COL_START];
3           PP [PP_W_LAST_WCB_COL]  = .OVERLAP [DCB_W_COL_START] +
3                                     .OVERLAP [DCB_W_NO_COLS] - 1;
3
3           PP [PP_L_MOVE_SIZE] = .OVERLAP [DCB_W_NO_ROWS] *
3                                 .OVERLAP [DCB_W_NO_COLS];
3
3           DCB_START_ROW = .OVERLAP [DCB_W_ROW_START] - .PP [PP_W_ROW] +1;
3           DCB_START_COL = .OVERLAP [DCB_W_COL_START] - .PP [PP_W_COL] +1;
3
3           PP [PP_W_FROM_INDEX] = (.DCB_START_ROW -1) *.DCB [DCB_W_NO_COLS]
3                                   + .DCB_START_COL - 1;
```

```
3
3    PP [PP_W_TO_INDEX] = (.OVERLAP [DCB_W_ROW_START] -1) *
3                        .WCB [WCB_W_NO_COLS] +
3                        .OVERLAP [DCB_B_COL_START] -1;
3
3    IF .DCB [DCB_V_BORDERED]
3    THEN
4        BEGIN             ! Bordered display
4        LOCAL
4            UPPER_ROW,          ! Row above top row of pasted display
4            LOWER_ROW,          ! Row below bottom row of pasted display
4            LEFT_COL,                   ! Col. to left of pasted display
4            RIGHT_COL,          ! Col. to right of pasted display
4            LDES : REF BLOCK [,BYTE];! Address of dynamic descr. in
4                                    ! DCB that points to label
4                                    ! string.
4
4        LDES = DCB [DCB_Q_LABEL_DESC];
4
4        !+
4        ! Compute the row and column numbers where the borders fall.
4        ! Note these rows and columns may not map into the buffer
4        ! and need to be validated before use.
4        !-
4        UPPER_ROW = .PP [PP_W_ROW] - 1 ;
4        LOWER_ROW = .PP [PP_W_ROW] + .DCB [DCB_W_NO_ROWS];
4        LEFT_COL  = .PP [PP_W_COL] - 1 ;
4        RIGHT_COL = .PP [PP_W_COL] + .DCB [DCB_W_NO_COLS];
4
4        IF .LDES [DSC$W_LENGTH] NEQ 0
4        THEN
5            BEGIN   ! Label position computation
5            CASE .DCB [DCB_B_LABEL_POS] FROM SMG$K_TOP TO SMG$K_RIGHT OF
5            SET
5                [SMG$K_TOP]:
6                    BEGIN   ! Label in top row
6                    IF .UPPER_ROW GEQ 1
6                    THEN
7                        BEGIN           ! Top row in buffer
7                        LOCAL
7                            DCOLS : SIGNED; ! Dest. col. start
7
7                        DCOLS =       .PP [PP_W_COL] +
7                                      .DCB [DCB_W_LABEL_UNITS] -2 ;
7
7                        IF .DCOLS LEQ .WCB [WCB_W_NO_COLS]
7                        THEN
8                            BEGIN   ! partially on screen
8                            LOCAL
8                                DCOLE : SIGNED; ! Dest. col end
8
8                            DCOLE = MIN ( (.LDES [DSC$W_LENGTH] + .DCOLS -1),
9                                      (.PP [PP_W_COL] +
8                                       .DCB [DCB_W_NO_COLS] ),
8                                       .WCB [WCB_W_NO_COLS]);
8
8                            PP [PP_W_LABEL_BYTES_TO_MOVE] =
                    MAX ( 0, .DCOLE +1 -
                             MAX (0, .DCOLS) );

            IF .PP [PP_W_COL] LEQ 0
            THEN
                BEGIN       ! Using tail end of label
                PP [PP_W_SRC_LABEL_OFF] =
                    .LDES [DSC$W_LENGTH] -
                    .PP [PP_W_LABEL_BYTES_TO_MOVE];
                END ! Using tail end of label
            ELSE
                BEGIN       ! Using front end of label
                PP [PP_W_SRC_LABEL_OFF] = 0;
                END;        ! Using front end of label

            PP [PP_W_DST_LABEL_OFF] = (.UPPER_ROW -1) *
                                .WCB [WCB_W_NO_COLS] +
                                MAX(0, .DCOLS - 1);
            END; ! Partially on screen

        END;        ! Top row in buffer
    END;    ! Label in top row

[SMG$K_BOTTOM]:
    BEGIN   ! Label in bottom row
    IF .LOWER_ROW LEQ .WCB [WCB_W_NO_ROWS]
    THEN
        BEGIN       ! Bottom row in buffer
        LOCAL
            DCOLS : SIGNED; ! Dest. col. start

        DCOLS =       .PP [PP_W_COL] +
                      .DCB [DCB_W_LABEL_UNITS] - 2;

        IF .DCOLS LEQ .WCB [WCB_W_NO_COLS]
        THEN
            BEGIN ! Partially visible
            LOCAL
                DCOLE : SIGNED; ! Dest. col. end

            DCOLE = MIN ( (.LDES [DSC$W_LENGTH] + .DCOLS -1),
                      (.PP [PP_W_COL] +
                       .DCB [DCB_W_NO_COLS] ),
                       .WCB [WCB_W_NO_COLS]);

            PP [PP_W_LABEL_BYTES_TO_MOVE] =
                    MAX ( 0, .DCOLE + 1 -
                             MAX (0, .DCOLS) );


            IF .PP [PP_W_COL] LEQ 0
            THEN
                BEGIN       ! Using tail end of label
                PP [PP_W_SRC_LABEL_OFF] =
                    .LDES [DSC$W_LENGTH] -
                    .PP [PP_W_LABEL_BYTES_TO_MOVE];
                END ! Using tail end of label
```

```
8               ELSE
9                   BEGIN       ! Using front end of label
9                   PP [PP_W_SRC_LABEL_OFF] = 0;
8                   END;        ! Using front end of label
8               PP [PP_W_DST_LABEL_OFF] = (.LOWER_ROW -1) *
8                                         .WCB [WCB_W_NO_COLS] +
8                                         MAX(0, .DCOLS - 1);
8
7               END ;! Partially visible
6           END;        ! Bottom row in buffer
5       END;    ! Label in bottom row
5
5   [SMG$K_LEFT]:
6       BEGIN   !Label in left column
6       IF .LEFT_COL GEQ 1
6       THEN
7           BEGIN       ! Left column in buffer
7           LOCAL
7               DROWS : SIGNED; ! Dest. row start
7
7           DROWS =       .PP [PP_W_ROW] +
7                         .DCB [DCB_W_LABEL_UNITS] - 2;
7
7           IF .DROWS LEQ .WCB [WCB_W_NO_ROWS]
7           THEN
8               BEGIN ! Partially visible
8               LOCAL
8                   DROWE : SIGNED ; ! Dest. row end
8
8               DROWE = MIN ( (.LDES [DSC$W_LENGTH] + .DROWS -1),
9                         (.PP [PP_W_ROW] +
8                          .DCB [DCB_W_NO_ROWS] ),
8                          .WCB [WCB_W_NO_ROWS]);
8
8               PP [PP_W_LABEL_BYTES_TO_MOVE] =
8                       MAX ( 0, .DROWE + 1 -
8                                MAX (0, .DROWS) );
8
8               IF .PP [PP_W_ROW] LEQ 0
8               THEN
9                   BEGIN       ! Using tail end of label
9                   PP [PP_W_SRC_LABEL_OFF] =
9                     .LDES [DSC$W_LENGTH] -
9                     .PP [PP_W_LABEL_BYTES_TO_MOVE];
9                   END ! Using tail end of label
8               ELSE
9                   BEGIN       ! Using front end of label
9                   PP [PP_W_SRC_LABEL_OFF] = 0;
8                   END;        ! Using front end of label
8
8               PP [PP_W_DST_LABEL_OFF] = (.DROWS -1) *
8                                         .WCB [WCB_W_NO_COLS] +
8                                         MAX(0, .LEFT_COL - 1);
7               END; ! Partially visible
6           END;        ! Left column in buffer
5       END;    ! Label in left column
5
```

```
5                       [SMG$K_RIGHT]:
6                           BEGIN   ! Label in right column
6                           IF .RIGHT_COL LEQ .WCB [WCB_W_NO_COLS]
6                           THEN
7                               BEGIN       ! Right column in buffer
7                               LOCAL
7                                   DROWS : SIGNED; ! Dest. row start
7
7                               DROWS =       .PP [PP_W_ROW] +
7                                             .DCB [DCB_W_LABEL_UNITS] - 2;
7
7                               IF .DROWS LEQ .WCB [WCB_W_NO_ROWS]
7                               THEN
8                                   BEGIN ! Partially visible
8                                   LOCAL
8                                     DROWE : SIGNED ; ! Dest. row end
8
8                                   DROWE = MIN ( (.LDES [DSC$W_LENGTH] + .DROWS -1),
9                                            (.PP [PP_W_ROW] +
8                                             .DCB [DCB_W_NO_ROWS] ),
8                                             .WCB [WCB_W_NO_ROWS]);
8
8                                   PP [PP_W_LABEL_BYTES_TO_MOVE] =
8                                           MAX ( 0, .DROWE + 1 -
8                                                    MAX (0, .DROWS) );
8
8
8                                   IF .PP [PP_W_ROW] LEQ 0
8                                   THEN
9                                       BEGIN       ! Using tail end of label
9                                       PP [PP_W_SRC_LABEL_OFF] =
9                                           .LDES [DSC$W_LENGTH] -
9                                           .PP [PP_W_LABEL_BYTES_TO_MOVE];
9                                       END ! Using tail end of label
8                                   ELSE
9                                       BEGIN       ! Using front end of label
9                                       PP [PP_W_SRC_LABEL_OFF] = 0;
8                                       END;        ! Using front end of label
8
8                                   PP [PP_W_DST_LABEL_OFF] = (.DROWS -1) *
8                                                       .WCB [WCB_W_NO_COLS] +
8                                                       MAX(0, .RIGHT_COL - 1);
8
7                                   END; ! Partially visible
6                               END;        ! Right column in buffer
5                           END;    ! Label in right column
5
5                       [OUTRANGE]:
5                           RETURN (SMG$_FATERRLIB);
5
5                   TES:
4                   END;    ! Label position computation
3               END;        ! Bordered display
2           END;    ! Overlap
2
2 !+
2 ! If the virtual display width matches the window control block width,
                                                                    E 5
SMG$DISPLAY_LIN SMG$DISPLAY_LINKS - Virtual Display Linkages    16-Sep-1984 00:29:22    VAX-11 Bliss-32 V4.0-742           Page 124
1-096           SMG$$CALC_PASTE_TRANSF - Calculate pasting tran 14-Sep-1984 13:09:43    [SMGRTL.SRC]SMGDISLIN.B32;1             (25)

; 4403         4640  2 ! and the width of the data to be move during a mapping operation is
; 4404         4641  2 ! this same width, then both the source area and the destination are
; 4405         4642  2 ! contiguous sets of bytes and can be moved with a single CH$MOVE.
; 4406         4643  2 ! If not, they have to be moved a row at a time since that is how many
; 4407         4644  2 ! are piece-wise contiguous.
; 4408         4645  2 !-
; 4409         4646  2     PP [PP_V_CONTIG] = 0; ! Assume not contiguous until so proven.
; 4410         4647  2     IF .DCB [DCB_W_NO_COLS]   EQL .WCB [WCB_W_NO_COLS]     AND
; 4411         4648  2        .PP [PP_W_MOVE_LENGTH] EQL .DCB [DCB_W_NO_COLS]
; 4412         4649  2     THEN
; 4413         4650  2         PP [PP_V_CONTIG] = 1;    ! Contiguous
; 4414         4651  2
; 4415         4652  2     RETURN (SS$_NORMAL);
; 4416         4653  1     END;                   ! End of routine SMG$$CALC_PASTE_TRANSF



                                              0FFC 00000          .ENTRY   SMG$$CALC_PASTE_TRANSF, Save R2,R3,R4,R5,-   ; 4299
                                                                           R6,R7,R8,R9,R10,R11                          ;
                             5E           2C   C2 00002           SUBL2    #44, SP                                      ;
                             57      04   AC   D0 00005           MOVL     PP, R7                                       ; 4350
                             50      14   A7   D0 00009           MOVL     20(R7), PBCB                                 ;
                             59      08   A0   D0 0000D           MOVL     8(PBCB), WCB                                 ; 4351
                             52      10   A7   D0 00011           MOVL     16(R7), DCB                                  ; 4352
                      18     AE      24   A7   9E 00015           MOVAB    36(R7), 24(SP)                               ; 4357
                                     18   BE   B4 0001A           CLRW     @24(SP)                                      ;
                      14     AE      26   A7   9E 0001D           MOVAB    38(R7), 20(SP)                               ; 4358
                                     14   BE   B4 00022           CLRW     @20(SP)                                      ;
                      10     AE      28   A7   9E 00025           MOVAB    40(R7), 16(SP)                               ; 4359
                                     10   BE   B4 0002A           CLRW     @16(SP)                                      ;
                      0C     AE      18   A7   9E 0002D           MOVAB    24(R7), 12(SP)                               ; 4362
                             50           62   3C 00032           MOVZWL   (DCB), R0                                    ;
                             51      0C   BE   32 00035           CVTWL    @12(SP), R1                                  ;
                             50           51   C0 00039           ADDL2    R1, R0                                       ;
        24    AE             50           01   A3 0003C           SUBW3    #1, R0, TEMP                                 ;
                      26     AE      02   A2   B0 00041           MOVW     2(DCB), TEMP+2                               ; 4363
                      08     AE      1A   A7   9E 00046           MOVAB    26(R7), 8(SP)                                ; 4364
                             50      04   A2   3C 0004B           MOVZWL   4(DCB), R0                                   ;
                             51      08   BE   32 0004F           CVTWL    @8(SP), R1                                   ;
                             50           51   C0 00053           ADDL2    R1, R0                                       ;
        28    AE             50           01   A3 00056           SUBW3    #1, R0, TEMP+4                               ;
                      04     AE      06   A2   3C 0005B           MOVZWL   6(DCB), 4(SP)                                ; 4365
                      2A     AE      04   AE   B0 00060           MOVW     4(SP), TEMP+6                                ;
                                     1C   AE   9F 00065           PUSHAB   OVERLAP                                      ; 4373
                                     28   AE   9F 00068           PUSHAB   TEMP                                         ;
                                          59   DD 0006B           PUSHL    WCB                                          ;
               00000000G     00           03   FB 0006D           CALLS    #3, SMG$$OCCLUDE                             ;
                             06           50   E8 00074           BLBS     R0, 2$                                       ;
                                     1C   A7   B4 00077           CLRW     28(R7)                                       ; 4379
                                        0258   31 0007A 1$:       BRW      42$                                          ; 4372
                      1C     A7      1E   AE   B0 0007D 2$:       MOVW     OVERLAP+2, 28(R7)                            ; 4393
                      22     A7      22   AE   B0 00082           MOVW     OVERLAP+6, 34(R7)                            ; 4394
                      2F     A7      1C   AE   B0 00087           MOVW     OVERLAP, 47(R7)                              ; 4396
                             50      1C   AE   3C 0008C           MOVZWL   OVERLAP, R0                                  ; 4398
                             51      1E   AE   3C 00090           MOVZWL   OVERLAP+2, R1                                ;
```

```
                                     50          51  C0 00094          ADDL2   R1, R0                                   ;
               31  A7                50          01  A3 00097          SUBW3   #1, R0, 49(R7)                           ;
                                     53     20   AE  3C 0009C          MOVZWL  OVERLAP+4, R3                            ; 4400
                              33     A7          53  B0 000A0          MOVW    R3, 51(R7)                               ;
                                     50     22   AE  3C 000A4          MOVZWL  OVERLAP+6, R0                            ; 4402
                                     51     FF A340  9E 000A8          MOVAB   -1(R3)[R0], R1                           ;
                              35     A7          51  B0 000AD          MOVW    R1, 53(R7)                               ;
                                     50     1E   AE  3C 000B1          MOVZWL  OVERLAP+2, R0                            ; 4405
                                     51     22   AE  3C 000B5          MOVZWL  OVERLAP+6, R1                            ;
               2B  A7                50          51  C5 000B9          MULL3   R1, R0, 43(R7)                           ;
                                     51     1C   AE  3C 000BE          MOVZWL  OVERLAP, R1                              ; 4407
                                     50     0C   BE  32 000C2          CVTWL   @12(SP), R0                              ;
                                     51          50  C2 000C6          SUBL2   R0, R1                                   ;
                                     50     08   BE  32 000C9          CVTWL   @8(SP), R0                               ; 4408
                   50                53          50  C3 000CD          SUBL3   R0, R3, R0                               ;
                                     51     04   AE  C4 000D1          MULL2   4(SP), R1                                ; 4410
                                     54        8041  9E 000D5          MOVAB   (DCB_START_COL)+[R1], R4                 ; 4411
                              1E     A7          54  B0 000D9          MOVW    R4, 30(R7)                               ;
                                     50     1C   AE  3C 000DD          MOVZWL  OVERLAP, R0                              ; 4413
                                                 50  D7 000E1          DECL    R0                                       ;
                                     5A     06   A9  3C 000E3          MOVZWL  6(WCB), R10                              ; 4414
                                     50          5A  C4 000E7          MULL2   R10, R0                                  ;
                                     51     FF A340  9E 000EA          MOVAB   -1(R3)[R0], R1                           ; 4415
                              20     A7          51  B0 000EF          MOVW    R1, 32(R7)                               ;
                                     83     2F   A2  E9 000F3          BLBC    47(DCB), 1$                              ; 4417
                                     50     08   A2  9E 000F7          MOVAB   8(R2), LDES                              ; 4429
                                     55     0C   BE  32 000FB          CVTWL   @12(SP), UPPER_ROW                       ; 4436
                                                 55  D7 000FF          DECL    UPPER_ROW                                ;
                                     5B     0C   BE  32 00101          CVTWL   @12(SP), R11                             ; 4437
                                     51     02   A2  3C 00105          MOVZWL  2(DCB), R1                               ;
                                     5B          51  C0 00109          ADDL2   R1, R11                                  ;
                                     51          5B  D0 0010C          MOVL    R11, LOWER_ROW                           ;
                                     54     08   BE  32 0010F          CVTWL   @8(SP), LEFT_COL                         ; 4438
                                                 54  D7 00113          DECL    LEFT_COL                                 ;
                                     58     08   BE  32 00115          CVTWL   @8(SP), R8                               ; 4439
                                     58     04   AE  C0 00119          ADDL2   4(SP), R8                                ;
                                     53          58  D0 0011D          MOVL    R8, RIGHT_COL                            ;
                                     56          60  3C 00120          MOVZWL  (LDES), R6                               ; 4441
                                                 74  13 00123          BEQL    12$                                      ;
                   03                00     31   A2  8F 00125          CASEB   49(DCB), #0, #3                          ; 4444
     0141          00D6            0071        0010     0012A 3$:      .WORD   4$-3$,-                                  ;
                                                                               13$-3$,-                                 ;
                                                                               23$-3$,-                                 ;
                                                                               33$-3$                                   ;
                                     50 00000000G  8F  D0 00132        MOVL    #SMG$_FATERRLIB, R0                      ; 4631
                                                 04     00139          RET                                              ;
                                                 55  D5 0013A 4$:      TSTL    UPPER_ROW                                ; 4448
                                                 5B  15 0013C          BLEQ    12$                                      ;
                                     50     08   BE  32 0013E          CVTWL   @8(SP), R0                               ; 4455
                                     6E     2C   A2  3C 00142          MOVZWL  44(DCB), (SP)                            ;
                                     50          6E  C0 00146          ADDL2   (SP), R0                                 ;
                                     50          02  C2 00149          SUBL2   #2, DCOLS                                ;
                                     5A          50  D1 0014C          CMPL    DCOLS, R10                               ; 4457
                                                 63  14 0014F          BGTR    14$                                      ;
                                     52     FF A046  9E 00151          MOVAB   -1(DCOLS)[R6], R2                        ; 4463
                                     58          52  D1 00156          CMPL    R2, R8                                   ; 4464
                                                 03  15 00159          BLEQ    5$                                       ;
```

```
                                                               G 5
SMG$DISPLAY_LIN SMG$DISPLAY_LINKS - Virtual Display Linkages    16-Sep-1984 00:29:22    VAX-11 Bliss-32 V4.0-742              Page 126
1-096           SMG$$CALC_PASTE_TRANSF - Calculate pasting tran 14-Sep-1984 13:09:43    [SMGRTL.SRC]SMGDISLIN.B32;1                (25)

                              52                58  D0 0015B          MOVL    R8, R2                                    :
                              5A                52  D1 0015E 5$:      CMPL    R2, R10                                   : 4466
                                                03  15 00161          BLEQ    6$                                        :
                              52                5A  D0 00163          MOVL    R10, R2                                   :
                              58                50  D0 00166 6$:      MOVL    DCOLS, R8                                 : 4470
                                                02  18 00169          BGEQ    7$                                        :
                                                58  D4 0016B          CLRL    R8                                        :
                              52                58  C2 0016D 7$:      SUBL2   R8, R2                                    :
                                                52  D6 00170          INCL    R2                                        : 4469
                                                02  18 00172          BGEQ    8$                                        :
                                                52  D4 00174          CLRL    R2                                        :
                         18   BE                52  B0 00176 8$:      MOVW    R2, @24(SP)                               :
                                        08      BE  B5 0017A          TSTW    @8(SP)                                    : 4472
                                                08  14 0017D          BGTR    9$                                        :
              14      BE      56        18      BE  A3 0017F          SUBW3   @24(SP), R6, @20(SP)                      : 4477
                                                03  11 00185          BRB     10$                                       : 4472
                                        14      BE  B4 00187 9$:      CLRW    @20(SP)                                   : 4481
                                                55  D7 0018A 10$:     DECL    R5                                        : 4484
                              55                5A  C4 0018C          MULL2   R10, R5                                   : 4485
                              02                50  F4 0018F          SOBGEQ  R0, 11$                                   : 4486
                                                50  D4 00192          CLRL    R0                                        :
              10      BE      55                50  A1 00194 11$:     ADDW3   R0, R5, @16(SP)                           :
                                                63  11 00199 12$:     BRB     22$                                       : 4444
    51        02      A9      10                00  ED 0019B 13$:     CMPZV   #0, #16, 2(WCB), LOWER_ROW                : 4494
                                                76  19 001A1          BLSS    24$                                       :
                              50        08      BE  32 001A3          CVTWL   @8(SP), R0                                : 4501
                              55        2C      A2  3C 001A7          MOVZWL  44(DCB), R5                               :
                              50                55  C0 001AB          ADDL2   R5, R0                                    :
                              50                02  C2 001AE          SUBL2   #2, DCOLS                                 :
                              5A                50  D1 001B1          CMPL    DCOLS, R10                                : 4503
                                                48  14 001B4 14$:     BGTR    22$                                       :
                              52        FF A046     9E 001B6          MOVAB   -1(DCOLS)[R6], R2                         : 4509
                              58                52  D1 001BB          CMPL    R2, R8                                    : 4510
                                                03  15 001BE          BLEQ    15$                                       :
                              52                58  D0 001C0          MOVL    R8, R2                                    :
                              5A                52  D1 001C3 15$:     CMPL    R2, R10                                   : 4512
                                                03  15 001C6          BLEQ    16$                                       :
                              52                5A  D0 001C8          MOVL    R10, R2                                   :
                              55                50  D0 001CB 16$:     MOVL    DCOLS, R5                                 : 4516
                                                02  18 001CE          BGEQ    17$                                       :
                                                55  D4 001D0          CLRL    R5                                        :
                              52                55  C2 001D2 17$:     SUBL2   R5, R2                                    :
                                                52  D6 001D5          INCL    R2                                        : 4515
                                                02  18 001D7          BGEQ    18$                                       :
                                                52  D4 001D9          CLRL    R2                                        :
                         18   BE                52  B0 001DB 18$:     MOVW    R2, @24(SP)                               :
                                        08      BE  B5 001DF          TSTW    @8(SP)                                    : 4519
                                                08  14 001E2          BGTR    19$                                       :
              14      BE      56        18      BE  A3 001E4          SUBW3   @24(SP), R6, @20(SP)                      : 4524
                                                03  11 001EA          BRB     20$                                       : 4519
                                        14      BE  B4 001EC 19$:     CLRW    @20(SP)                                   : 4528
                                                51  D7 001EF 20$:     DECL    R1                                        : 4530
                              51                5A  C4 001F1          MULL2   R10, R1                                   : 4531
                              02                50  F4 001F4          SOBGEQ  R0, 21$                                   : 4532
                                                50  D4 001F7          CLRL    R0                                        :
              10      BE      51                50  A1 001F9 21$:     ADDW3   R0, R1, @16(SP)                           :
                                                69  11 001FE 22$:     BRB     32$                                       : 4444
```

```
                                                                  H  5
SMG$DISPLAY_LIN SMG$DISPLAY_LINKS - Virtual Display Linkages    16-Sep-1984 00:29:22     VAX-11 Bliss-32 V4.0-742            Page 127
1-096           SMG$$CALC_PASTE_TRANSF - Calculate pasting tran 14-Sep-1984 13:09:43     [SMGRTL.SRC]SMGDISLIN.B32;1             (25)

                                                     54  D5 00200 23$:    TSTL    LEFT_COL                                    ; 4540
                                                     65  15 00202         BLEQ    32$
                                       51       0C   BE  32 00204         CVTWL   @12(SP), R1                                 ; 4547
                                       50       2C   A2  3C 00208         MOVZWL  44(DCB), R0
                                       51            50  C0 0020C         ADDL2   R0, R1
                                       50       FE   A1  9E 0020F         MOVAB   -2(R1), DROWS
           50       02   A9            10            00  ED 00213         CMPZV   #0, #16, 2(WCB), DROWS                      ; 4549
                                                     6A  19 00219 24$:    BLSS    34$
                                       51       FF A046  9E 0021B         MOVAB   -1(DROWS)[R6], R1                           ; 4555
                                       5B            51  D1 00220         CMPL    R1, R11                                     ; 4556
                                                     03  15 00223         BLEQ    25$
                                       51            5B  D0 00225         MOVL    R11, R1
           51       02   A9            10            00  ED 00228 25$:    CMPZV   #0, #16, 2(WCB), R1                         ; 4558
                                                     04  18 0022E         BGEQ    26$
                                       51       02   A9  3C 00230         MOVZWL  2(WCB), R1
                                       51            50  7D 00234 26$:    MOVQ    DROWS, R1                                   ; 4562
                                                     51  D5 00237         TSTL    R1
                                                     02  18 00239         BGEQ    27$
                                                     51  D4 0023B         CLRL    R1
                                       52            51  C2 0023D 27$:    SUBL2   R1, R2
                                                     52  D6 00240         INCL    R2                                          ; 4561
                                                     02  18 00242         BGEQ    28$
                                                     52  D4 00244         CLRL    R2
                                  18   BE            52  B0 00246 28$:    MOVW    R2, @24(SP)
                                                0C   BE  B5 0024A         TSTW    @12(SP)                                     ; 4564
                                                     08  14 0024D         BGTR    29$
                    14   BE            56       18   BE  A3 0024F         SUBW3   @24(SP), R6, @20(SP)                        ; 4569
                                                     03  11 00255         BRB     30$                                         ; 4564
                                                14   BE  B4 00257 29$:    CLRW    @20(SP)                                     ; 4573
                                                     50  D7 0025A 30$:    DECL    R0                                          ; 4576
                                       50            5A  C4 0025C         MULL2   R10, R0                                     ; 4577
                                       02            54  F4 0025F         SOBGEQ  R4, 31$                                     ; 4578
                                                     54  D4 00262         CLRL    R4
                    10   BE            50            54  A1 00264 31$:    ADDW3   R4, R0, @16(SP)
                                                     6A  11 00269 32$:    BRB     42$                                         ; 4444
                                       5A            53  D1 0026B 33$:    CMPL    RIGHT_COL, R10                              ; 4585
                                                     65  14 0026E         BGTR    42$
                                       51       0C   BE  32 00270         CVTWL   @12(SP), R1                                 ; 4592
                                       50       2C   A2  3C 00274         MOVZWL  44(DCB), R0
                                       51            50  C0 00278         ADDL2   R0, R1
                                       50       FE   A1  9E 0027B         MOVAB   -2(R1), DROWS
           50       02   A9            10            00  ED 0027F         CMPZV   #0, #16, 2(WCB), DROWS                      ; 4594
                                                     4E  19 00285 34$:    BLSS    42$
                                       51       FF A046  9E 00287         MOVAB   -1(DROWS)[R6], R1                           ; 4600
                                       5B            51  D1 0028C         CMPL    R1, R11                                     ; 4601
                                                     03  15 0028F         BLEQ    35$
                                       51            5B  D0 00291         MOVL    R11, R1
           51       02   A9            10            00  ED 00294 35$:    CMPZV   #0, #16, 2(WCB), R1                         ; 4603
                                                     04  18 0029A         BGEQ    36$
                                       51       02   A9  3C 0029C         MOVZWL  2(WCB), R1
                                       51            50  7D 002A0 36$:    MOVQ    DROWS, R1                                   ; 4607
                                                     51  D5 002A3         TSTL    R1
                                                     02  18 002A5         BGEQ    37$
                                                     51  D4 002A7         CLRL    R1
                                       52            51  C2 002A9 37$:    SUBL2   R1, R2
                                                     52  D6 002AC         INCL    R2                                          ; 4606
                                                     02  18 002AE         BGEQ    38$
```

```
                                                                  I  5
SMG$DISPLAY_LIN SMG$DISPLAY_LINKS - Virtual Display Linkages     16-Sep-1984 00:29:22    VAX-11 Bliss-32 V4.0-742          Page 128
1-096           SMG$$CALC_PASTE_TRANSF - Calculate pasting tran 14-Sep-1984 13:09:43    [SMGRTL.SRC]SMGDISLIN.B32;1          (25)

                                          52  D4 002B0        CLRL     R2                                                 ;
                       18   BE            52  B0 002B2 38$:   MOVW     R2, @24(SP)                                        ;
                                    0C    BE  B5 002B6        TSTW     @12(SP)                                            ; 4610
                                          08  14 002B9        BGTR     39$                                                ;
             14   BE        56      18    BE  A3 002BB        SUBW3    @24(SP), R6, @20(SP)                               ; 4615
                                          03  11 002C1        BRB      40$                                                ; 4610
                                    14    BE  B4 002C3 39$:   CLRW     @20(SP)                                            ; 4619
                                          50  D7 002C6 40$:   DECL     R0                                                 ; 4622
                            50            5A  C4 002C8        MULL2    R10, R0                                            ; 4623
                            02            53  F4 002CB        SOBGEQ   R3, 41$                                            ; 4624
                                          53  D4 002CE        CLRL     R3                                                 ;
             10   BE        50            53  A1 002D0 41$:   ADDW3    R3, R0, @16(SP)                                    ;
                       2A   A7            02  8A 002D5 42$:   BICB2    #2, 42(R7)                                         ; 4646
                       04   AE      06    A9  B1 002D9        CMPW     6(WCB), 4(SP)                                      ; 4647
                                          0B  12 002DE        BNEQ     43$                                                ;
                       04   AE      22    A7  B1 002E0        CMPW     34(R7), 4(SP)                                      ; 4648
                                          04  12 002E5        BNEQ     43$                                                ;
                       2A   A7            02  88 002E7        BISB2    #2, 42(R7)                                         ; 4650
                            50            01  D0 002EB 43$:   MOVL     #1, R0                                             ; 4652
                                              04 002EE        RET                                                         ; 4653

; Routine Size:  751 bytes,    Routine Base:  _SMG$CODE + 166C


; 4417          4654  1 !<BLF/PAGE>
```

```
 1 %SBTTL 'SMG$$CHECK_OCCLUSION - Check pastings for occlusion'
 1 GLOBAL ROUTINE SMG$$CHECK_OCCLUSION (
 1                                   PBCB : REF $PBCB_DECL
 1                                       ) =
 1 !++
 1 ! FUNCTIONAL DESCRIPTION:
 1 !
 1 !       This procedure checks the overlap in all the pastings to
 1 !       a given pasteboard -- setting a bit in the respective pasting
 1 !       packets to record whether each pasting is occluded by
 1 !       higher (more-recently pasted) pastings.  This procedure is
 1 !       invoked whenever there is a change in the pasting order or
 1 !       pasting position of any virtual display.
 1 !       If we can determine at pasting time that a particular virtual
 1 !       display (as pasted) and with a given succession of higher-pasted
 1 !       virtual displays is not occluded by any of them,
 1 !       we can make mapping run faster since a change to a virtual
 1 !       display which is not occluded causes only the changed virtual
 1 !       display to be remapped.  Higher pasted virtual displays do not
 1 !       have to be remapped since it is known that they do not occlude
 1 !       the changed one.
 1 !
 1 ! CALLING SEQUENCE:
 1 !
 1 !       ret_status.wlc.v = SMG$$CHECK_OCCLUSION ( PBCB.rab.r)
 1 !
 1 ! FORMAL PARAMETERS:
 1 !
 1 !       PBCB.rab.r              Address of pasteboard control block
 1 !                               which has something new or different
 1 !                               pasted to it.
 1 !
 1 ! IMPLICIT INPUTS:
 1 !
 1 !       NONE
 1 !
 1 ! IMPLICIT OUTPUTS:
 1 !
 1 !       NONE
 1 !
 1 ! COMPLETION STATUS:
 1 !
 1 !       SS$_NORMAL      Normal successful completion
 1 !
 1 ! SIDE EFFECTS:
 1 !
 1 !       NONE
 1 !--
 1
 2     BEGIN
 2     LOCAL
 2         THIS_PP : REF $PP_DECL,
 2                         ! Addr of pasting packet for upper-most pasted
 2                         ! virtual display.
 2
 2         THIS_Q_HEAD : REF BLOCK [,BYTE];
 2                         ! Addr of 2 longwords that form queue header in
```

```
2                           ! PP currently under inspection.
2
2   !+
2   ! To initialize for the rest of the algorithm, run through whole pasting
2   ! list marking all packets not occluded.
2   !-
2       THIS_Q_HEAD = .PBCB [PBCB_A_PP_NEXT];         ! 1st (more recent pasting)
2       WHILE .THIS_Q_HEAD NEQ PBCB [PBCB_A_PP_NEXT]
2       DO
3           BEGIN   ! Init. pass
3           THIS_PP = .THIS_Q_HEAD - PP_PBCB_QUEUE_OFFSET; ! To top of packet
3           THIS_PP [ PP_V_OCCLUDED] = 0;   ! Init to not occluded
3           THIS_Q_HEAD = .THIS_PP [PP_A_NEXT_PBCB];     ! To queue header in
3                                                        !  next packet
2           END;    ! Init. pass
2
2       THIS_Q_HEAD = .PBCB [PBCB_A_PP_NEXT];
2       THIS_PP = .THIS_Q_HEAD - PP_PBCB_QUEUE_OFFSET;
2
2   !+
2   ! Loop for all pasting packets starting with most-recently pas:ed one.
2   !-
2       WHILE .THIS_Q_HEAD NEQ PBCB [PBCB_A_PP_NEXT]
2       DO
3           BEGIN   ! For all displays from top to bottom
3           LOCAL
3               NEXT_PP         : REF $PP_DECL,
3                                   ! Addr of pasting packet currently under
3                                   ! inspection.
3
3               NEXT_PP_Q_HEAD : REF BLOCK [,BYTE],
3                                   ! Addr of 2 longwords that form queue
3                                   ! header in PP currently under
3                                   ! inspection.
3
3               TEMP_THIS : BLOCK [8,BYTE],
3                                   ! Area of projection of THIS virtual
3                                   ! display on pasteboard
3
3               TEMP_NEXT : BLOCK [8,BYTE],
3                                   ! Area of projection of NEXT virtual
3                                   ! display on pasteboard
3
3               THIS_DCB : REF BLOCK [,BYTE];
3                                   ! Addr of virtual display currently
3                                   ! under inspection.
3
3           !+
3           ! Recalculate pasting packet address and DCB address for this
3           ! iteration.
3           !-
3           THIS_PP = .THIS_Q_HEAD - PP_PBCB_QUEUE_OFFSET;
3           THIS_DCB = .THIS_PP [PP_A_DCB_ADDR];
3
3           !+
3           ! It is safe to assume that there is at least one virtual
3           ! display pasted to this pasteboard -- but there may not be more
```

```
; 4533          4769 3             ! than one.  Be careful about reaching ahead to a packet that
; 4534          4770 3             ! may not be a packet.  If doesn't exist, pointer will be
; 4535          4771 3             ! pointing back into PBCB -- and inner loop will not be
; 4536          4772 3             ! executed.
; 4537          4773 3             !-
; 4538          4774 3             NEXT_PP_Q_HEAD = .THIS_PP [PP_A_NEXT_PBCB];
; 4539          4775 3
; 4540          4776 3             IF .NEXT_PP_Q_HEAD NEQ PBCB [PBCB_A_PP_NEXT]
; 4541          4777 3             THEN
; 4542          4778 4                 BEGIN          ! NEXT exists
; 4543          4779 4                 NEXT_PP = .NEXT_PP_Q_HEAD - PP_PBCB_QUEUE_OFFSET;
; 4544          4780 4                 !+
; 4545          4781 4                 ! Form a representation of the projection of THIS virtual
; 4546          4782 4                 ! display onto pasteboard coordinate system.
; 4547          4783 4                 !-
; 4548          4784 4                 TEMP_THIS [DCB_W_ROW_START] = .THIS_DCB [DCB_W_ROW_START] +
; 4549          4785 4                                               .THIS_PP [PP_W_ROW] - 1;
; 4550          4786 4                 TEMP_THIS [DCB_W_NO_ROWS]   = .THIS_DCB [DCB_W_NO_ROWS];
; 4551          4787 4                 TEMP_THIS [DCB_W_COL_START] = .THIS_DCB [DCB_W_COL_START] +
; 4552          4788 4                                               .THIS_PP [PP_W_COL] - 1;
; 4553          4789 4                 TEMP_THIS [DCB_W_NO_COLS]   = .THIS_DCB [DCB_W_NO_COLS];
; 4554          4790 4
; 4555          4791 4                 !+
; 4556          4792 4                 ! If this virtual display is bordered, its projection is
; 4557          4793 4                 ! bigger than if it were not.  Adjust its projection
; 4558          4794 4                 ! representation.
; 4559          4795 4                 !-
; 4560          4796 4                 IF .THIS_DCB [DCB_V_BORDERED]
; 4561          4797 4                 THEN
; 4562          4798 5                     BEGIN   ! Border adjustment
; 4563          4799 5                     TEMP_THIS [DCB_W_ROW_START] = .TEMP_THIS [DCB_W_ROW_START] - 1;
; 4564          4800 5                     TEMP_THIS [DCB_W_NO_ROWS]   = .TEMP_THIS [DCB_W_NO_ROWS] + 2;
; 4565          4801 5                     TEMP_THIS [DCB_W_COL_START] = .TEMP_THIS [DCB_W_COL_START] - 1;
; 4566          4802 5                     TEMP_THIS [DCB_W_NO_COLS]   = .TEMP_THIS [DCB_W_NO_COLS] + 2;
; 4567          4803 4                     END;    ! Border adjustment
; 4568          4804 4
; 4569          4805 3                 END;          ! Next exists
; 4570          4806 3
; 4571          4807 3             WHILE .NEXT_PP_Q_HEAD NEQ PBCB [PBCB_A_PP_NEXT]
; 4572          4808 3             DO
; 4573          4809 4                 BEGIN         ! For all displays from current to bottom
; 4574          4810 4                 LOCAL
; 4575          4811 4                     NEXT_DCB : REF $DCB_DECL,
; 4576          4812 4                                     ! Addr of DCB associated with NEXT_PP
; 4577          4813 4                     OVERLAP : BLOCK [8,BYTE];
; 4578          4814 4                                     ! Returned by SMG$$OCCLUDE, but not
; 4579          4815 4                                     ! used in this context
; 4580          4816 4
; 4581          4817 4                 NEXT_PP = .NEXT_PP_Q_HEAD - PP_PBCB_QUEUE_OFFSET;
; 4582          4818 4                 NEXT_DCB = .NEXT_PP [PP_A_DCB_ADDR];
; 4583          4819 4
; 4584          4820 4                 !+
; 4585          4821 4                 ! Form a representation of the projection of NEXT virtual
; 4586          4822 4                 ! display onto pasteboard coordinate system.
; 4587          4823 4                 !-
; 4588          4824 4                 TEMP_NEXT [DCB_W_ROW_START] = .NEXT_DCB [DCB_W_ROW_START] +
; 4589          4825 4                                               .NEXT_PP [PP_W_ROW] - 1;
; 4590          4826  4                 TEMP_NEXT [DCB_W_NO_ROWS]   = .NEXT_DCB [DCB_W_NO_ROWS];
; 4591          4827  4                 TEMP_NEXT [DCB_W_COL_START] = .NEXT_DCB [DCB_W_COL_START] +
; 4592          4828  4                                               .NEXT_PP [PP_W_COL] - 1;
; 4593          4829  4                 TEMP_NEXT [DCB_W_NO_COLS]   = .NEXT_DCB [DCB_W_NO_COLS];
; 4594          4830  4
; 4595          4831  4                 !+
; 4596          4832  4                 ! If this next virtual display is bordered, its projection is
; 4597          4833  4                 ! bigger than if it were not.  Adjust its projection
; 4598          4834  4                 ! representation.
; 4599          4835  4                 !-
; 4600          4836  4                 IF .NEXT_DCB [DCB_V_BORDERED]
; 4601          4837  4                 THEN
; 4602          4838  5                     BEGIN   ! Border adjustment
; 4603          4839  5                     TEMP_NEXT [DCB_W_ROW_START] = .TEMP_NEXT [DCB_W_ROW_START] - 1;
; 4604          4840  5                     TEMP_NEXT [DCB_W_NO_ROWS]   = .TEMP_NEXT [DCB_W_NO_ROWS] + 2;
; 4605          4841  5                     TEMP_NEXT [DCB_W_COL_START] = .TEMP_NEXT [DCB_W_COL_START] - 1;
; 4606          4842  5                     TEMP_NEXT [DCB_W_NO_COLS]   = .TEMP_NEXT [DCB_W_NO_COLS] + 2;
; 4607          4843  4                     END;     ! Border adjustment
; 4608          4844  4
; 4609          4845  4                 !+
; 4610          4846  4                 ! Check to see if THIS virtual display occludes NEXT
; 4611          4847  4                 ! vitual display and if so set occlusion bit of NEXT.
; 4612          4848  4                 !-
; 4613          4849  4                 IF SMG$$OCCLUDE ( TEMP_NEXT, TEMP_THIS, OVERLAP)
; 4614          4850  4                 THEN
; 4615          4851  4                     NEXT_PP [PP_V_OCCLUDED] = 1;
; 4616          4852  4
; 4617          4853  4                 !+
; 4618          4854  4                 ! Walk chain in direction of earlier pasted packets.
; 4619          4855  4                 !-
; 4620          4856  4                 NEXT_PP_Q_HEAD = .NEXT_PP [PP_A_NEXT_PBCB];
; 4621          4857  4
; 4622          4858  3                 END;         ! For all displays from current to bottom
; 4623          4859  3
; 4624          4860  3             !+
; 4625          4861  3             ! Walk chain in direction of earlier pasted packets.
; 4626          4862  3             !-
; 4627          4863  3             THIS_Q_HEAD = .THIS_PP [PP_A_NEXT_PBCB];
; 4628          4864  3
; 4629          4865  2             END;    ! For all displays from top to bottom
; 4630          4866  2
; 4631          4867  2         RETURN (SS$_NORMAL);
; 4632          4868  2
; 4633          4869  1         END;                    ! End of routine SMG$$CHECK_OCCLUSION



                                  00FC 00000          .ENTRY  SMG$$CHECK_OCCLUSION, Save R2,R3,R4,R5,R6,-  ; 4656
                                                              R7                                           ;
                     5E          18  C2 00002         SUBL2   #24, SP                                      ;
                     56      04  AC  D0 00005         MOVL    PBCB, R6                                     ; 4718
                     54          66  D0 00009         MOVL    (R6), THIS_Q_HEAD                            ;
                     56          54  D1 0000C  1$:    CMPL    THIS_Q_HEAD, R6                              ; 4719
                                 0E  13 0000F         BEQL    2$                                           ;
                     52      F8  A4  9E 00011         MOVAB   -8(R4), THIS_PP                              ; 4722
```

```
                              2A   A2        01   8A 00015       BICB2   #1, 42(THIS_PP)                          : 4723
                                   54   08   A2   D0 00019       MOVL    8(THIS_PP), THIS_Q_HEAD                  : 4724
                                             ED   11 0001D       BRB     1$                                       : 4719
                                   56        66   D0 0001F 2$:   MOVL    (R6), THIS_Q_HEAD                        : 4728
                                   52   F8   A4   9E 00022       MOVAB   -8(R4), THIS_PP                          : 4729
                                   56        54   D1 00026 3$:   CMPL    THIS_Q_HEAD, R6                          : 4734
                                             03   12 00029       BNEQ    4$
                                           00BB   31 0002B       BRW     9$
                                   52   F8   A4   9E 0002E 4$:   MOVAB   -8(R4), THIS_PP                          : 4763
                                   50   10   A2   D0 00032       MOVL    16(THIS_PP), THIS_DCB                    : 4764
                                   55   08   A2   D0 00036       MOVL    8(THIS_PP), NEXT_PP_Q_HEAD               : 4774
                                   56        55   D1 0003A       CMPL    NEXT_PP_Q_HEAD, R6                       : 4776
                                             3F   13 0003D       BEQL    5$
                                   53   F8   A5   9E 0003F       MOVAB   -8(R5), NEXT_PP                          : 4779
                                   51        60   3C 00043       MOVZWL  (THIS_DCB), R1                           : 4785
                                   57   18   A2   32 00046       CVTWL   24(THIS_PP), R7
                                   51        57   C0 0004A       ADDL2   R7, R1
                  10   AE          51        01   A3 0004D       SUBW3   #1, R1, TEMP_THIS
                              12   AE   02   A0   B0 00052       MOVW    2(THIS_DCB), TEMP_THIS+2                 : 4786
                                   51   04   A0   3C 00057       MOVZWL  4(THIS_DCB), R1                          : 4788
                                   57   1A   A2   32 0005B       CVTWL   26(THIS_PP), R7
                                   51        57   C0 0005F       ADDL2   R7, R1
                  14   AE          51        01   A3 00062       SUBW3   #1, R1, TEMP_THIS+4
                              16   AE   06   A0   B0 00067       MOVW    6(THIS_DCB), TEMP_THIS+6                 : 4789
                                   0E   2F   A0   E9 0006C       BLBC    47(THIS_DCB), 5$                         : 4796
                                        10   AE   B7 00070       DECW    TEMP_THIS                                : 4799
                              12   AE        02   A0 00073       ADDW2   #2, TEMP_THIS+2                          : 4800
                                        14   AE   B7 00077       DECW    TEMP_THIS+4                              : 4801
                              16   AE        02   A0 0007A       ADDW2   #2, TEMP_THIS+6                          : 4802
                                   56        55   D1 0007E 5$:   CMPL    NEXT_PP_Q_HEAD, R6                       : 4807
                                             5F   13 00081       BEQL    8$
                                   53   F8   A5   9E 00083       MOVAB   -8(R5), NEXT_PP                          : 4817
                                   50   10   A3   D0 00087       MOVL    16(NEXT_PP), NEXT_DCB                    : 4818
                                   51        60   3C 0008B       MOVZWL  (NEXT_DCB), R1                           : 4825
                                   57   18   A3   32 0008E       CVTWL   24(NEXT_PP), R7
                                   51        57   C0 00092       ADDL2   R7, R1
                  08   AE          51        01   A3 00095       SUBW3   #1, R1, TEMP_NEXT
                              0A   AE   02   A0   B0 0009A       MOVW    2(NEXT_DCB), TEMP_NEXT+2                 : 4826
                                   51   04   A0   3C 0009F       MOVZWL  4(NEXT_DCB), R1                          : 4828
                                   57   1A   A3   32 000A3       CVTWL   26(NEXT_PP), R7
                                   51        57   C0 000A7       ADDL2   R7, R1
                  0C   AE          51        01   A3 000AA       SUBW3   #1, R1, TEMP_NEXT+4
                              0E   AE   06   A0   B0 000AF       MOVW    6(NEXT_DCB), TEMP_NEXT+6                 : 4829
                                   0E   2F   A0   E9 000B4       BLBC    47(NEXT_DCB), 6$                         : 4836
                                        08   AE   B7 000B8       DECW    TEMP_NEXT                                : 4839
                              0A   AE        02   A0 000BB       ADDW2   #2, TEMP_NEXT+2                          : 4840
                                        0C   AE   B7 000BF       DECW    TEMP_NEXT+4                              : 4841
                              0E   AE        02   A0 000C2       ADDW2   #2, TEMP_NEXT+6                          : 4842
                                             5E   DD 000C6 6$:   PUSHL   SP                                       : 4849
                                        14   AE   9F 000C8       PUSHAB  TEMP_THIS
                                        10   AE   9F 000CB       PUSHAB  TEMP_NEXT
                       00000000G   00        03   FB 000CE       CALLS   #3, SMG$$OCCLUDE
                                   04        50   E9 000D5       BLBC    R0, 7$
                              2A   A3        01   88 000D8       BISB2   #1, 42(NEXT_PP)                          : 4851
                                   55   08   A3   D0 000DC 7$:   MOVL    8(NEXT_PP), NEXT_PP_Q_HEAD               : 4856
                                             9C   11 000E0       BRB     5$                                       : 4807
                                   54   08   A2   D0 000E2 8$:   MOVL    8(THIS_PP), THIS_Q_HEAD                  : 4863
```

```
                                          FF3D  31 000E6          BRW      3$                                                 ; 4734
                                  50        01  D0 000E9 9$:      MOVL     #1, R0                                             ; 4867
                                                04 000EC          RET                                                         ; 4869

; Routine Size:  237 bytes,    Routine Base:  _SMG$CODE + 195B

; 4634          4870  1 !<BLF/PAGE>
```

```
1 %SBTTL 'SMG$$CHECK_OCCLUSION_FIRST - Check pastings for occlusion'
1 GLOBAL ROUTINE SMG$$CHECK_OCCLUSION_FIRST (
1                                       PBCB : REF $PBCB_DECL
1                                            ) =
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This procedure updates the overlap bit in all the pastings to
1 !       a given pasteboard -- setting a bit in the respective pasting
1 !       packets to record whether each pasting is occluded by
1 !       the highest (most-recently pasted) pasting.  This procedure is
1 !       invoked whenever a virtual display is freshly pasted.
1 !
1 !       This routine differs from SMG$$CHECK_OCCLUSION it that that
1 !       routine compares all pastings to all other pasting.  This
1 !       routine only compares the top pasting to all the others since
1 !       the addition of this new top one can only add an occlusion to
1 !       a lower one.  Any occlusions already existing at a low level
1 !       cannot have been modified by pasting one more on top and there
1 !       is no reason to recalculate their relationship to each other.
1 !
1 !       If we can determine at pasting time that a particular virtual
1 !       display (as pasted) and with a given succession of higher-pasted
1 !       virtual displays is not occluded by any of them,
1 !       we can make mapping run faster since a change to a virtual
1 !       display which is not occluded causes only the changed virtual
1 !       display to be remapped.  Higher pasted virtual displays do not
1 !       have to be remapped since it is known that they do not occlude
1 !       the changed one.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       ret_status.wlc.v = SMG$$CHECK_OCCLUSION_FIRST ( PBCB.rab.r)
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       PBCB.rab.r              Address of pasteboard control block
1 !                               which has a new virtual display pasted
1 !                               to it.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !       SS$_NORMAL      Normal successful completion
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !--
1
```

```
2       BEGIN
2       LOCAL
2           THIS_PP : REF $PP_DECL,
2                               ! Addr of pasting packet for upper-most pasted
2                               ! virtual display.
2
2           THIS_Q_HEAD : REF BLOCK [,BYTE],
2                               ! Addr of 2 longwords that form queue header in
2                               ! PP currently under inspection.
2
2           NEXT_PP : REF $PP_DECL,
2                               ! Addr of pasting packet currently under
2                               ! inspection.
2
2           NEXT_PP_Q_HEAD : REF BLOCK [,BYTE],
2                               ! Addr of 2 longwords that form queue
2                               ! header in PP currently under
2                               ! inspection.
2
2           TEMP_THIS : BLOCK [8,BYTE],
2                               ! Area of projection of THIS virtual
2                               ! display on pasteboard
2
2           TEMP_NEXT : BLOCK [8,BYTE],
2                               ! Area of projection of NEXT virtual
2                               ! display on pasteboard
2
2           THIS_DCB : REF $DCB_DECL;
2
2                               ! Addr of virtual display currently
2                               ! under inspection.
2
2       THIS_Q_HEAD = .PBCB [PBCB_A_PP_NEXT];          ! Most recent pasting
2       THIS_PP = .THIS_Q_HEAD - PP_PBCB_QUEUE_OFFSET;
2
2       THIS_DCB = .THIS_PP [PP_A_DCB_ADDR];
2
2   !+
2   ! It is safe to assume that there is at least one virtual
2   ! display pasted to this pasteboard -- but there may not be more than
2   ! one.  Be careful about reaching ahead to a packet that may not be a
2   ! packet.  If doesn't exist, pointer will be pointing back into PBCB
2   ! -- and inner loop will not be executed.
2   !-
2       NEXT_PP_Q_HEAD = .THIS_PP [PP_A_NEXT_PBCB];
2
2       IF .NEXT_PP_Q_HEAD NEQ PBCB [PBCB_A_PP_NEXT]
2       THEN
3           BEGIN   ! NEXT exists
3           NEXT_PP = .NEXT_PP_Q_HEAD - PP_PBCB_QUEUE_OFFSET;
3           !+
3           ! Form a representation of the projection of THIS virtual
3           ! display onto pasteboard coordinate system.
3           !-
3           TEMP_THIS [DCB_W_ROW_START] = .THIS_DCB [DCB_W_ROW_START] +
3                                         .THIS_PP [PP_W_ROW] - 1;
3           TEMP_THIS [DCB_W_NO_ROWS]   = .THIS_DCB [DCB_W_NO_ROWS];
```

```
3               TEMP_THIS [DCB_W_COL_START] = .THIS_DCB [DCB_W_COL_START] +
3                                             .THIS_PP [PP_W_COL] - 1;
3               TEMP_THIS [DCB_W_NO_COLS]   = .THIS_DCB [DCB_W_NO_COLS];
3
3               !+
3               ! If this virtual display is bordered, its projection is bigger
3               ! than if it were not.  Adjust its projection representation.
3               !-
3               IF .THIS_DCB [DCB_V_BORDERED]
3               THEN
4                   BEGIN       ! Border adjustment
4                   TEMP_THIS [DCB_W_ROW_START] = .TEMP_THIS [DCB_W_ROW_START] - 1;
4                   TEMP_THIS [DCB_W_NO_ROWS]   = .TEMP_THIS [DCB_W_NO_ROWS] + 2;
4                   TEMP_THIS [DCB_W_COL_START] = .TEMP_THIS [DCB_W_COL_START] - 1;
4                   TEMP_THIS [DCB_W_NO_COLS]   = .TEMP_THIS [DCB_W_NO_COLS] + 2;
3                   END;        ! Border adjustment
3
2               END;    ! Next exists
2
2           WHILE .NEXT_PP_Q_HEAD NEQ PBCB [PBCB_A_PP_NEXT]
2           DO
3               BEGIN   ! For all displays from current to bottom
3               LOCAL
3                   NEXT_DCB : REF $DCB_DECL,
3                                       ! Addr of DCB associated with NEXT_PP
3                   OVERLAP : BLOCK [8,BYTE];
3                                       ! Returned by SMG$$OCCLUDE, but not
3                                       ! used in this context
3
3               NEXT_PP = .NEXT_PP_Q_HEAD - PP_PBCB_QUEUE_OFFSET;
3               NEXT_DCB = .NEXT_PP [PP_A_DCB_ADDR];
3
3               !+
3               ! Form a representation of the projection of NEXT virtual
3               ! display onto pasteboard coordinate system.
3               !-
3               TEMP_NEXT [DCB_W_ROW_START] = .NEXT_DCB [DCB_W_ROW_START] +
3                                             .NEXT_PP [PP_W_ROW] - 1;
3               TEMP_NEXT [DCB_W_NO_ROWS]   = .NEXT_DCB [DCB_W_NO_ROWS];
3               TEMP_NEXT [DCB_W_COL_START] = .NEXT_DCB [DCB_W_COL_START] +
3                                             .NEXT_PP [PP_W_COL] - 1;
3               TEMP_NEXT [DCB_W_NO_COLS]   = .NEXT_DCB [DCB_W_NO_COLS];
3
3                   !+
3                   ! If this next virtual display is bordered, its projection is
3                   ! bigger than if it were not.  Adjust its projection
3                   ! representation.
3                   !-
3                   IF .NEXT_DCB [DCB_V_BORDERED]
3                   THEN
4                       BEGIN   ! Border adjustment
4                       TEMP_NEXT [DCB_W_ROW_START] = .TEMP_NEXT [DCB_W_ROW_START] - 1;
4                       TEMP_NEXT [DCB_W_NO_ROWS]   = .TEMP_NEXT [DCB_W_NO_ROWS] + 2;
4                       TEMP_NEXT [DCB_W_COL_START] = .TEMP_NEXT [DCB_W_COL_START] - 1;
4                       TEMP_NEXT [DCB_W_NO_COLS]   = .TEMP_NEXT [DCB_W_NO_COLS] + 2;
3                       END;    ! Border adjustment
3
```

```
; 4807          5042  3             !+
; 4808          5043  3             ! Check to see if THIS virtual display occludes NEXT
; 4809          5044  3             ! vitual display and if so set occlusion bit of NEXT.
; 4810          5045  3             !-
; 4811          5046  3             IF SMG$$OCCLUDE ( TEMP_NEXT, TEMP_THIS, OVERLAP)
; 4812          5047  3             THEN
; 4813          5048  3                 NEXT_PP [PP_V_OCCLUDED] = 1;
; 4814          5049  3
; 4815          5050  3             !+
; 4816          5051  3             ! Walk chain in direction of earlier pasted packets.
; 4817          5052  3             !-
; 4818          5053  3             NEXT_PP_Q_HEAD = .NEXT_PP [PP_A_NEXT_PBCB];
; 4819          5054  3
; 4820          5055  2             END;     ! For all displays from current to bottom
; 4821          5056  2
; 4822          5057  2
; 4823          5058  2         RETURN (SS$_NORMAL);
; 4824          5059  2
; 4825          5060  1         END;                    ! End of routine SMG$$CHECK_OCCLUSION_FIRST



                                           003C 00000          .ENTRY  SMG$$CHECK_OCCLUSION_FIRST, Save R2,R3,R4,-  ; 4872
                                                                       R5                                           :
                      5E           18   C2 00002          SUBL2   #24, SP                                       :
                      51     04    BC   D0 00005          MOVL    @PBCB, THIS_Q_HEAD                            : 4960
                      51           08   C2 00009          SUBL2   #8, THIS_PP                                   : 4961
                      50     10    A1   D0 0000C          MOVL    16(THIS_PP), THIS_DCB                         : 4963
                      53     08    A1   D0 00010          MOVL    8(THIS_PP), NEXT_PP_Q_HEAD                    : 4972
                 04   AC           53   D1 00014          CMPL    NEXT_PP_Q_HEAD, PBCB                          : 4974
                                   3F   13 00018          BEQL    1$                                            :
                      52     F8    A3   9E 0001A          MOVAB   -8(R3), NEXT_PP                               : 4977
                      54     60        3C 0001E          MOVZWL  (THIS_DCB), R4                                : 4983
                      55     18    A1   32 00021          CVTWL   24(THIS_PP), R5                               :
                      54           55   C0 00025          ADDL2   R5, R4                                        :
  10    AE            54           01   A3 00028          SUBW3   #1, R4, TEMP_THIS                             :
                 12   AE     02    A0   B0 0002D          MOVW    2(THIS_DCB), TEMP_THIS+2                      : 4984
                      54     04    A0   3C 00032          MOVZWL  4(THIS_DCB), R4                               : 4986
                      51     1A    A1   32 00036          CVTWL   26(THIS_PP), R1                               :
                      51           54   C0 0003A          ADDL2   R4, R1                                        :
  14    AE            51           01   A3 0003D          SUBW3   #1, R1, TEMP_THIS+4                           :
                 16   AE     06    A0   B0 00042          MOVW    6(THIS_DCB), TEMP_THIS+6                      : 4987
                      0E     2F    A0   E9 00047          BLBC    47(THIS_DCB), 1$                              : 4993
                             10    AE   B7 0004B          DECW    TEMP_THIS                                     : 4996
                 12   AE           02   A0 0004E          ADDW2   #2, TEMP_THIS+2                               : 4997
                             14    AE   B7 00052          DECW    TEMP_THIS+4                                   : 4998
                 16   AE           02   A0 00055          ADDW2   #2, TEMP_THIS+6                               : 4999
                 04   AC           53   D1 00059 1$:      CMPL    NEXT_PP_Q_HEAD, PBCB                          : 5004
                                   5F   13 0005D          BEQL    4$                                            :
                      52     F8    A3   9E 0005F          MOVAB   -8(R3), NEXT_PP                               : 5014
                      50     10    A2   D0 00063          MOVL    16(NEXT_PP), NEXT_DCB                         : 5015
                      51     60        3C 00067          MOVZWL  (NEXT_DCB), R1                                : 5022
                      54     18    A2   32 0006A          CVTWL   24(NEXT_PP), R4                               :
                      51           54   C0 0006E          ADDL2   R4, R1                                        :
  08    AE            51           01   A3 00071          SUBW3   #1, R1, TEMP_NEXT                             :
```

```
                                                                   G  6
SMG$DISPLAY_LIN SMG$DISPLAY_LINKS - Virtual Display Linkages     16-Sep-1984 00:29:22    VAX-11 Bliss-32 V4.0-742          Page 139
1-096           SMG$CHECK_OCCLUSION_FIRST - Check pastings for 14-Sep-1984 13:09:43    [SMGRTL.SRC]SMGDISLIN.B32;1         (27)

                              0A   AE      02   A0  B0 00076          MOVW    2(NEXT_DCB), TEMP_NEXT+2                   ; 5023
                                   51      04   A0  3C 0007B          MOVZWL  4(NEXT_DCB), R1                            ; 5025
                                   54      1A   A2  32 0007F          CVTWL   26(NEXT_PP), R4                            ;
                                   51           54  C0 00083          ADDL2   R4, R1                                     ;
                  0C    AE         51           01  A3 00086          SUBW3   #1, R1, TEMP_NEXT+4                        ;
                              0E   AE      06   A0  B0 0008B          MOVW    6(NEXT_DCB), TEMP_NEXT+6                   ; 5026
                                   0E      2F   A0  E9 00090          BLBC    47(NEXT_DCB), 2$                           ; 5033
                                           08   AE  B7 00094          DECW    TEMP_NEXT                                  ; 5036
                              0A   AE           02  A0 00097          ADDW2   #2, TEMP_NEXT+2                            ; 5037
                                           0C   AE  B7 0009B          DECW    TEMP_NEXT+4                                ; 5038
                              0E   AE           02  A0 0009E          ADDW2   #2, TEMP_NEXT+6                            ; 5039
                                                5E  DD 000A2  2$:     PUSHL   SP                                         ; 5046
                                           14   AE  9F 000A4          PUSHAB  TEMP_THIS                                  ;
                                           10   AE  9F 000A7          PUSHAB  TEMP_NEXT                                  ;
                       00000000G   00           03  FB 000AA          CALLS   #3, SMG$$OCCLUDE                           ;
                                   04           50  E9 000B1          BLBC    R0, 3$                                     ;
                              2A   A2           01  88 000B4          BISB2   #1, 42(NEXT_PP)                            ; 5048
                                   53      08   A2  D0 000B8  3$:     MOVL    8(NEXT_PP), NEXT_PP_Q_HEAD                 ; 5053
                                                9B  11 000BC          BRB     1$                                         ; 5004
                              50                01  D0 000BE  4$:     MOVL    #1, R0                                     ; 5058
                                                    04 000C1          RET                                                ; 5060

; Routine Size:  194 bytes,    Routine Base:  _SMG$CODE + 1A48


; 4826          5061  1 !<BLF/PAGE>
```

```
%SBTTL 'SMG$CREATE_PASTEBOARD - Create Pasteboard Control Block (PBCB)'
GLOBAL ROUTINE SMG$CREATE_PASTEBOARD (
                                ROWS,
                                COLS,
                                PBCB_ADDR
                                    ) =
!++
! FUNCTIONAL DESCRIPTION:
!
!       This routine allocates space for a pasteboard control block.
!       It also allocates a buffer called the output buffer
!       which is used to buffer output to this terminal
!       (if buffering is enabled).
!
! CALLING SEQUENCE:
!
!       ret_status.wlc.v = SMG$CREATE_PASTEBOARD (
!                                               ROWS.rl.r,
!                                               COLS.rl.r,
!                                               PBCB.wl.r)
!
! FORMAL PARAMETERS:
!
!       ROWS.rl.r       Max. number of rows that a window onto this
!                       pasteboard will have.
!
!       COLS.rl.r       Max. number of columns that a window onto this
!                       pasteboard will have.
!
!       PBCB_ADDR.wl.r  Address of the newly-created PBCB -- returned to
!                       caller.
!
! IMPLICIT INPUTS:
!
!       NONE
!
! IMPLICIT OUTPUTS:
!
!       items in PBCB get filled in.
!       in particular, an output buffer is allocated.
!
! COMPLETION STATUS:
!
!       SS$_NORMAL      Normal successful completion
!       LIB$_INSVIRMEM  Insufficient virtual memory to allocate needed
!                       buffers.
!
! SIDE EFFECTS:
!
!       NONE
!--
```

```
2        BEGIN
2        LOCAL
2            PBCB : REF $PBCB_DECL, ! Address of PBCB allocated.
2
2            STATUS;          ! Status of subroutine calls
2
2        LITERAL
2
2            PBCB_K_OUTBUF_DEFAULT_SIZE
2
2                = 256;       ! Default size for output buffer
2                             ! (if all other algorithms fail)
2
2    !+
2    ! Allocate space for the PBCB itself.
2    !-
3        IF NOT (STATUS = LIB$GET_VM (%REF (PBCB_K_SIZE), PBCB))
2        THEN
2            RETURN (.STATUS);
2
2        CH$FILL (0, PBCB_K_SIZE, .PBCB);     ! Clear all fields to default 0
2
2    !+
2    ! Allocate the window control block that goes along with this
2    ! pasteboard, returning failure if we can't.
2    !-
3        IF NOT (STATUS = SMG$$CREATE_WCB (.ROWS, .COLS, PBCB [PBCB_A_WCB]))
2        THEN
3            BEGIN   ! No more space
3            !+
3            ! If we can't get space for WCB, we might as well give back
3            ! the PBCB space itself.
3            !-
3            LIB$FREE_VM (%REF (PBCB_K_SIZE), PBCB);
3            RETURN (.STATUS);
3            END;    ! No more space
2
2    !+
2    ! Allocate output buffer that goes along with this pasteboard, returning
2    ! failure if we can't.
2    ! This buffer is used (if buffering is enabled) to buffer all output to
2    ! this terminal.
2    ! When V3B comes out, we should do a better job in figuring
2    ! out a good size for this buffer by looking at sysgen paramaters
2    ! and user quotas, etc.  For now, we just allocate a fixed space.
2    !-
2
2        STATUS = LIB$GET_VM (%REF (PBCB_K_OUTBUF_DEFAULT_SIZE),
2                             PBCB [PBCB_A_OUTPUT_BUFFER] );
2        IF NOT .STATUS
2        THEN
3            BEGIN
3            !+
3            ! If we can't get space for the output buffer, we might as well
3            ! give back the PBCB space itself as well as the WCB space.
3            ! Ignore any errors that occur while trying to free this space.
3            !-
```

```
; 4937          5170  3             SMG$DEALLOCATE_WCB( .PBCB [PBCB_A_WCB] );
; 4938          5171  3             LIB$FREE_VM (%REF (PBCB_K_SIZE), PBCB);
; 4939          5172  3             RETURN (.STATUS);
; 4940          5173  2             END;
; 4941          5174  2
; 4942          5175  2      PBCB [PBCB_W_OUTPUT_BUFSIZ] = PBCB_K_OUTBUF_DEFAULT_SIZE;    ! allocation
; 4943          5176  2
; 4944          5177  2   !+
; 4945          5178  2   ! Initialize pasting queue header to self.
; 4946          5179  2   !-
; 4947          5180  2      PBCB [PBCB_A_PP_NEXT] = PBCB [PBCB_A_PP_NEXT];
; 4948          5181  2      PBCB [PBCB_A_PP_PREV] = PBCB [PBCB_A_PP_NEXT];
; 4949          5182  2
; 4950          5183  2   !+
; 4951          5184  2   ! Initialize mode settings to default.
; 4952          5185  2   !-
; 4953          5186  2      PBCB [PBCB_L_MODE_SETTINGS] = PBCB_K_DEF_MODE_SETTINGS;
; 4954          5187  2
; 4955          5188  2   !+
; 4956          5189  2   ! Return the address of the PBCB we've built.
; 4957          5190  2   !-
; 4958          5191  2      .PBCB_ADDR = .PBCB;
; 4959          5192  2
; 4960          5193  2      RETURN (SS$_NORMAL);
; 4961          5194  1      END;                          ! Routine SMG$CREATE_PASTEBOARD



                                              00FC 00000          .ENTRY  SMG$CREATE_PASTEBOARD, Save R2,R3,R4,R5,-   ; 5063
                                                                          R6,R7                                        ;
                          57 00000000G  00  9E 00002              MOVAB   LIB$GET_VM, R7                               ;
                          5E            08  C2 00009              SUBL2   #8, SP                                       ;
                                    04  AE  9F 0000C              PUSHAB  PBCB                                         ; 5129
                   04  AE         014C  8F  3C 0000F              MOVZWL  #332, 4(SP)                                  ;
                                    04  AE  9F 00015              PUSHAB  4(SP)                                        ;
                          67            02  FB 00018              CALLS   #2, LIB$GET_VM                               ;
                          56            50  D0 0001B              MOVL    R0, STATUS                                   ;
                          51            56  E9 0001E              BLBC    STATUS, 2$                                   ;
   014C   8F          00  6E            00  2C 00021              MOVC5   #0, (SP), #0, #332, @PBCB                    ; 5133
                                    04  BE     00028                                                                   ;
              7E   04  AE            08  C1 0002A              ADDL3   #8, PBCB, -(SP)                              ; 5139
                          7E        04  AC  7D 0002F              MOVQ    ROWS, -(SP)                                  ;
                   0000V  CF            03  FB 00033              CALLS   #3, SMG$CREATE_WCB                           ;
                          56            50  D0 00038              MOVL    R0, STATUS                                   ;
                          21            56  E9 0003B              BLBC    STATUS, 1$                                   ;
                          52        04  AE  D0 0003E              MOVL    PBCB, R2                                     ; 5161
                                    6C  A2  9F 00042              PUSHAB  108(R2)                                      ;
                   04  AE         0100  8F  3C 00045              MOVZWL  #256, 4(SP)                                  ; 5160
                                    04  AE  9F 0004B              PUSHAB  4(SP)                                        ;
                          67            02  FB 0004E              CALLS   #2, LIB$GET_VM                               ; 5161
                          56            50  D0 00051              MOVL    R0, STATUS                                   ;
                          1F            56  E8 00054              BLBS    STATUS, 3$                                   ; 5162
                                    08  A2  DD 00057              PUSHL   8(R2)                                        ; 5170
                   0000V  CF            01  FB 0005A              CALLS   #1, SMG$DEALLOCATE_WCB                       ;
                                    04  AE  9F 0005F  1$:         PUSHAB  PBCB                                         ; 5171
```

```
                                                                   K  6
SMG$DISPLAY_LIN SMG$DISPLAY_LINKS - Virtual Display Linkages     16-Sep-1984 00:29:22    VAX-11 Bliss-32 V4.0-742            Page 143
1-096           SMG$$CREATE_PASTEBOARD - Create Pasteboard Cont 14-Sep-1984 13:09:43    [SMGRTL.SRC]SMGDISLIN.B32;1              (29)

                           04   AE     014C   8F  3C 00062          MOVZWL  #332, 4(SP)                                  ;
                                         04   AE  9F 00068          PUSHAB  4(SP)                                        ;
                    00000000G   00            02  FB 0006B          CALLS   #2, LIB$FREE_VM                              ;
                                50            56  D0 00072 2$:      MOVL    STATUS, R0                                   ; 5172
                                                  04 00075          RET                                                  ;
                                50       04   AE  D0 00076 3$:      MOVL    PBCB, R0                                     ; 5175
                           70   A0     0100   8F  B0 0007A          MOVW    #256, 112(R0)                                ;
                                60            50  D0 00080          MOVL    R0, (R0)                                     ; 5180
                           04   A0            50  D0 00083          MOVL    R0, 4(R0)                                    ; 5181
                           0C   A0            02  D0 00087          MOVL    #2, 12(R0)                                   ; 5186
                           0C   BC            50  D0 0008B          MOVL    R0, @PBCB_ADDR                               ; 5191
                                50            01  D0 0008F          MOVL    #1, R0                                       ; 5193
                                                  04 00092          RET                                                  ; 5194

; Routine Size:  147 bytes,    Routine Base:  _SMG$CODE + 1B0A


; 4962          5195  1 !<BLF/PAGE>
```

```
1 %SBTTL 'SMG$$CREATE_VIRTUAL_DISPLAY - Create Virtual Display'
1 GLOBAL ROUTINE SMG$$CREATE_VIRTUAL_DISPLAY (
1                                             NUM_ROWS,      ! height
1                                             NUM_COLS,      ! width
1                                             NEW_DISPLAY_ID,
1                                             DISPLAY_ATTRIBUTES,
1                                             VIDEO_ATTRIBUTES,
1                                             CHAR_SET
1                                             ) =
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine creates a new virtual display -- returning its
1 !       assigned display_id.  Its initial contents are blanks with
1 !       video attributes set to those specified.  The cursor
1 !       will be at row 1 column 1.
1 !       This is the inner-most create_virtual_display routine.  It
1 !       assumes all of its parameters are present.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       ret_status.wlc.v = SMG$$CREATE_VIRTUAL_DISPLAY (
1 !                               NUM_ROWS.rl.r,          ! Height
1 !                               NUM_COLS.rl.r,          ! Width
1 !                               NEW_DISPLAY_ID.wl.r,
1 !                               DISPLAY_ATTRIBUTES.rl.r,
1 !                               VIDEO_ATTRIBUTES.rl.r,
1 !                               CHAR_SET.rl.r)
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       NUM_ROWS.rl.r    Number of rows in new virtual display.
1 !
1 !       NUM_COLS.rl.r    Number of columns in new virtual display.
1 !
1 !       NEW_DISPLAY_ID.wl.r     Virtual display id of newly-created
1 !                               virtual display.
1 !
1 !       DISPLAY_ATTRIBUTES.rl.r The default display attributes.
1 !
1 !                       SMG$M_BORDER if virtual display is to be
1 !                               displayed with a border.
1 !
1 !                       SMG$M_TRUNC_ICON if an icon should be displayed
1 !                               when text overflows the display
1 !                               bounds.
1 !
1 !                       SMG$M_DISPLAY_CONTROLS if carriage controls (CR, LF,
1 !                               FF, VT, HT) should be displayed instead
1 !                               of executed.
1 !
1 !       VIDEO_ATTRIBUTES.rl.r   The default rendition code to be
1 !                       applied to all output to this display unless
1 !                       overridden on a particular output call.
1 !
1 !                       Values:
1 !
```

```
1 !                               SMG$M_BLINK     displays characters blinking.
1 !
1 !                               SMG$M_BOLD      displays characters in
1 !                                               higher-than-normal intensity.
1 !
1 !                               SMG$M_REVERSE   displays characters in reverse
1 !                                               video -- that is, using the
1 !                                               opposite default rendition of
1 !                                               the virtual display.
1 !
1 !                               SMG$M_UNDERLINE displays characters underlined.
1 !
1 !         CHAR_SET.rb.r         Specifies the default character set to be used
1 !                               for this display.
1 !                               Recognized values are:
1 !                                               SMG$C_UNITED_KINGDOM
1 !                                               SMG$C_ASCII (default)
1 !                                               SMG$C_SPEC_GRAPHICS
1 !                                               SMG$C_ALT_CHAR
1 !                                               SMG$C_ALT_GRAPHICS
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !         NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !         NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !         SS$_NORMAL            Normal successful completion
1 !         LIB$_INSVIRMEM        Insufficient virtual memory to allocate needed
1 !                               buffer.
1 !         SMG$_INVARG           Unrecognized Video Attributes
1 !                            or Unrecognized Display Attributes
1 !
1 ! SIDE EFFECTS:
1 !
1 !         NONE
1 !--
2     BEGIN
2     LOCAL
2         STATUS,                       ! Status of subroutine calls
2         DCB : REF $DCB_DECL,          ! Addr of display control block
2         DESC : REF BLOCK [8,BYTE];    ! Pointer to dynamic descriptor in
2                                       ! DCB for border label
2
2 !+
2 ! Allocate space for DCB itself.  Quit if we can't get it.
2 !-
2     IF NOT (STATUS = LIB$GET_VM ( %REF (DCB_K_SIZE), DCB))
2     THEN
2         RETURN (.STATUS);
2
2     CH$FILL (0, DCB_K_SIZE, .DCB);         ! set all fields to default of 0
2
```

```
2 !+
2 ! Set up dimensions of display
2 !-
2     DCB [DCB_W_ROW_START] = 1;
2     DCB [DCB_W_NO_ROWS]   = ..NUM_ROWS;
2     DCB [DCB_W_COL_START] = 1;
2     DCB [DCB_W_NO_COLS]   = ..NUM_COLS;
2     DCB [DCB_L_BUFSIZE]   = ..NUM_ROWS * ..NUM_COLS;
2
2 !+
2 ! Record default display attributes, default video attributes and
2 ! default character set.
2 !-
2     DCB [DCB_B_DEF_DISPLAY_ATTR] = ..DISPLAY_ATTRIBUTES;
2     DCB [DCB_B_DEF_VIDEO_ATTR]   = ..VIDEO_ATTRIBUTES;
2     DCB [DCB_B_DEF_CHAR_SET]     = ..CHAR_SET;
2
2 !+
2 ! Set up various fields in the DCB
2 !-
2     DCB [DCB_L_DID] = .DCB;                 ! Display id itself -- the
2                                             ! address of the DCB
2
2     DCB [DCB_W_CURSOR_ROW] = 1;             ! Cursor row and column to home
2     DCB [DCB_W_CURSOR_COL] = 1;
2
2     DCB [DCB_B_STRUCT_TYPE]  = DCB_K_STRUCT_TYPE; ! Mark as being a DCB
2     DCB [DCB_W_DCB_LENGTH]   = DCB_K_SIZE;        ! Size of structure
2
2     DCB [DCB_W_TOP_OF_SCRREG] = .DCB [DCB_W_ROW_START];
2     DCB [DCB_W_BOTTOM_OF_SCRREG] = .DCB [DCB_W_NO_ROWS];
2                                             ! init scrolling region
2 !+
2 ! Allocate enough space for both the text buffer and the attribute
2 ! buffer.
2 !-
3     IF NOT (STATUS = LIB$GET_VM (%REF (.DCB [DCB_L_BUFSIZE] *2),
3                                  DCB [DCB_A_TEXT_BUF]))
2     THEN
3         BEGIN   ! Ran out of space
3         !+
3         ! If we can't get enough space for the buffers, we might as well
3         ! give back the DCB space itself.
3         !-
3         LIB$FREE_VM ( %REF (DCB_K_SIZE), DCB); ! Return DCB space
3         RETURN (.STATUS);          ! Return the LIB$_INSVIRMEM from the
3                                    ! buffer allocation attempt
2         END;    ! Ran out of space
2
2 !+
2 ! Use upper half of space allocated as the attribute buffer.
2 !-
2     DCB [DCB_A_ATTR_BUF] = .DCB [DCB_A_TEXT_BUF] + .DCB [DCB_L_BUFSIZE];
2
2 !+
2 ! Initialize text and attribute buffers.
2 !-
```

```
2       CH$FILL (%C' ', .DCB [DCB_L_BUFSIZE], .DCB [DCB_A_TEXT_BUF]);
2       CH$FILL (.DCB [DCB_B_DEF_VIDEO_ATTR], .DCB [DCB_L_BUFSIZE],
2               .DCB [DCB_A_ATTR_BUF]);
2
2   !+
2   ! If we are dealing with a non-standard character set, allocate the
2   ! char_set buffer.  If we can't, bail out, giving back all the space
2   ! allocated on this transaction.
2   !-
2       IF .DCB [DCB_B_DEF_CHAR_SET] NEQ 0
2       THEN
3           BEGIN   ! Will need char_set buffer
4           IF NOT (STATUS = LIB$GET_VM ( DCB [DCB_L_BUFSIZE],
4                                         DCB [DCB_A_CHAR_SET_BUF]))
3           THEN
4               BEGIN       ! Bailout
4               !+
4               ! If we can't get space for buffer we need, give back the
4               ! the text and attribute buffer, and DCB itself before
4               ! quitting.
4               !-
4               LIB$FREE_VM (%REF (2 * .DCB [DCB_L_BUFSIZE]),
4                            DCB [DCB_A_TEXT_BUF]);
4               LIB$FREE_VM (%REF (DCB_K_SIZE), DCB);
4               RETURN (.STATUS);
3               END;        ! Bailout
3
3           CH$FILL (.DCB [DCB_B_DEF_CHAR_SET], .DCB [DCB_L_BUFSIZE],
3                    .DCB [DCB_A_CHAR_SET_BUF]);
3
2           END;    ! Will need char_set buffer
2
2
2   !+
2   ! Allocate and clear the line characteristics vector.
2   !-
3       IF NOT (STATUS = LIB$GET_VM ( %REF ( .DCB [DCB_W_NO_ROWS] +1),
3                                     DCB [DCB_A_LINE_CHAR]))
2       THEN
3           BEGIN   ! Error path
3           !+
3           ! Give back all space accumulated on this trans. before
3           ! bailing out.
3           !-
3           LIB$FREE_VM (%REF (2 * .DCB [DCB_L_BUFSIZE]),
3                        DCB [DCB_A_TEXT_BUF]);
3
3           IF .DCB [DCB_A_CHAR_SET_BUF] NEQ 0
3           THEN
3               LIB$FREE_VM ( DCB [DCB_L_BUFSIZE], DCB [DCB_A_CHAR_SET_BUF]);
3
3           LIB$FREE_VM (%REF (DCB_K_SIZE), DCB);
3
3           RETURN (.STATUS);
2           END;    ! Error path
2
2       CH$FILL ( 0, .DCB [DCB_W_NO_ROWS] + 1, .DCB [DCB_A_LINE_CHAR]);
```

```
; 5192          5424  2
; 5193          5425  2 !+
; 5194          5426  2 ! Initialize pasting queue headers to point to self.
; 5195          5427  2 !-
; 5196          5428  2     DCB [DCB_A_PP_NEXT] = DCB [DCB_A_PP_NEXT];
; 5197          5429  2     DCB [DCB_A_PP_PREV] = DCB [DCB_A_PP_NEXT];
; 5198          5430  2
; 5199          5431  2 !+
; 5200          5432  2 ! Initialize border label descriptor to virgin dynamic string descriptor
; 5201          5433  2 !-
; 5202          5434  2     DESC = DCB [DCB_Q_LABEL_DESC];
; 5203          5435  2
; 5204          5436  2     DESC [DSC$B_CLASS]          = DSC$K_CLASS_D;
; 5205          5437  2     DESC [DSC$B_DTYPE]          = DSC$K_DTYPE_T;
; 5206          5438  2
; 5207          5439  2 !+
; 5208          5440  2 ! Return the new display id to caller
; 5209          5441  2 !-
; 5210          5442  2     .NEW_DISPLAY_ID = .DCB;
; 5211          5443  2
; 5212          5444  2     RETURN (SS$_NORMAL);
; 5213          5445  1     END;                        ! Routine SMG$$CREATE_VIRTUAL_DISPLAY



                                          03FC 00000          .ENTRY  SMG$$CREATE_VIRTUAL_DISPLAY, Save R2,R3,R4,-; 5197
                                                                      R5,R6,R7,R8,R9                                 :
                        59 00000000G  00   9E 00002          MOVAB   LIB$FREE_VM, R9                                :
                        58 00000000G  00   9E 00009          MOVAB   LIB$GET_VM, R8                                 :
                        5E            08   C2 00010          SUBL2   #8, SP                                         :
                                 04   AE   9F 00013          PUSHAB  DCB                                            ; 5304
                  04    AE       70   8F   9A 00016          MOVZBL  #112, 4(SP)                                    :
                                 04   AE   9F 0001B          PUSHAB  4(SP)                                          :
                        68            02   FB 0001E          CALLS   #2, LIB$GET_VM                                 :
                        57            50   D0 00021          MOVL    R0, STATUS                                     :
                        03            57   E8 00024          BLBS    STATUS, 1$                                     :
                                    00F5   31 00027          BRW     7$                                             :
                        56       04   AE   D0 0002A 1$:      MOVL    DCB, R6                                        ; 5308
  0070   8F        00   6E            00   2C 0002E          MOVC5   #0, (SP), #0, #112, (R6)                       :
                                      66      00035                                                                 :
                        66            01   B0 00036          MOVW    #1, (R6)                                       ; 5313
                  02    A6       04   BC   B0 00039          MOVW    @NUM_ROWS, 2(R6)                               ; 5314
                  04    A6            01   B0 0003E          MOVW    #1, 4(R6)                                      ; 5315
                  06    A6       08   BC   B0 00042          MOVW    @NUM_COLS, 6(R6)                               ; 5316
        3C    A6  04    BC       08   BC   C5 00047          MULL3   @NUM_COLS, @NUM_ROWS, 60(R6)                   ; 5317
                  2F    A6       10   BC   90 0004E          MOVB    @DISPLAY_ATTRIBUTES, 47(R6)                    ; 5323
                  2E    A6       14   BC   90 00053          MOVB    @VIDEO_ATTRIBUTES, 46(R6)                      ; 5324
                  30    A6       18   BC   90 00058          MOVB    @CHAR_SET, 48(R6)                              ; 5325
                  38    A6            56   D0 0005D          MOVL    R6, 56(R6)                                     ; 5330
                  28    A6 00010001   8F   D0 00061          MOVL    #65537, 40(R6)                                 ; 5333
                  44    A6            11   90 00069          MOVB    #17, 68(R6)                                    ; 5336
                  45    A6       70   8F   9B 0006D          MOVZBW  #112, 69(R6)                                   ; 5337
                  48    A6            66   D0 00072          MOVL    (R6), 72(R6)                                   ; 5339
                                 10   A6   9F 00076          PUSHAB  16(R6)                                         ; 5347
        04    AE  3C    A6            01   78 00079          ASHL    #1, 60(R6), 4(SP)                              ; 5346
```

```
                                                                     D  7
SMG$DISPLAY_LIN SMG$DISPLAY_LINKS - Virtual Display Linkages    16-Sep-1984 00:29:22    VAX-11 Bliss-32 V4.0-742          Page 149
1-096           SMG$$CREATE_VIRTUAL_DISPLAY - Create Virtual Di 14-Sep-1984 13:09:43    [SMGRTL.SRC]SMGDISLIN.B32;1          (30)

                                           04   AE  9F 0007F         PUSHAB  4(SP)                                        ;
                                    68          02  FB 00082         CALLS   #2, LIB$GET_VM                               ; 5347
                                    57          50  D0 00085         MOVL    R0, STATUS                                   ;
                                    03          57  E8 00088         BLBS    STATUS, 2$                                   ;
                                         0083       31 0008B         BRW     6$                                           ;
                                    56     04   AE  D0 0008E 2$:     MOVL    DCB, R6                                      ; 5362
                 14    A6     10    A6     3C   A6  C1 00092         ADDL3   60(R6), 16(R6), 20(R6)                       ;
    3C    A6           20           6E     00   2C     00099         MOVC5   #0, (SP), #32, 60(R6), @16(R6)               ; 5367
                                           10   B6     0009F
    3C    A6     2E    A6           6E     00   2C     000A1         MOVC5   #0, (SP), 46(R6), 60(R6), @20(R6)            ; 5369
                                           14   B6     000A8
                                           30   A6  95 000AA         TSTB    48(R6)                                       ; 5376
                                                2A  13 000AD         BEQL    4$                                           ;
                                           18   A6  9F 000AF         PUSHAB  24(R6)                                       ; 5380
                                           3C   A6  9F 000B2         PUSHAB  60(R6)                                       ; 5379
                                    68          02  FB 000B5         CALLS   #2, LIB$GET_VM                               ; 5380
                                    57          50  D0 000B8         MOVL    R0, STATUS                                   ;
                                    0E          57  E8 000BB         BLBS    STATUS, 3$                                   ;
                                           10   A6  9F 000BE         PUSHAB  16(R6)                                       ; 5389
                 04    AE     3C    A6          01  78 000C1         ASHL    #1, 60(R6), 4(SP)                            ; 5388
                                           04   AE  9F 000C7         PUSHAB  4(SP)                                        ;
                                                42  11 000CA         BRB     5$                                           ; 5389
                                    50     04   AE  D0 000CC 3$:     MOVL    DCB, R0                                      ; 5394
    3C    A0     30    A0           6E          00  2C 000D0         MOVC5   #0, (SP), 48(R0), 60(R0), @24(R0)            ; 5395
                                           18   B0     000D7
                                    52     04   AE  D0 000D9 4$:     MOVL    DCB, R2                                      ; 5404
                                           4C   A2  9F 000DD         PUSHAB  76(R2)                                       ;
                              04    AE     02   A2  3C 000E0         MOVZWL  2(R2), 4(SP)                                 ; 5403
                                           04   AE  D6 000E5         INCL    4(SP)                                        ;
                                           04   AE  9F 000E8         PUSHAB  4(SP)                                        ;
                                    68          02  FB 000EB         CALLS   #2, LIB$GET_VM                               ; 5404
                                    57          50  D0 000EE         MOVL    R0, STATUS                                   ;
                                    2F          57  E8 000F1         BLBS    STATUS, 8$                                   ;
                                           10   A2  9F 000F4         PUSHAB  16(R2)                                       ; 5412
                 04    AE     3C    A2          01  78 000F7         ASHL    #1, 60(R2), 4(SP)                            ; 5411
                                           04   AE  9F 000FD         PUSHAB  4(SP)                                        ;
                                    69          02  FB 00100         CALLS   #2, LIB$FREE_VM                              ; 5412
                                           18   A2  D5 00103         TSTL    24(R2)                                       ; 5414
                                                09  13 00106         BEQL    6$                                           ;
                                           18   A2  9F 00108         PUSHAB  24(R2)                                       ; 5416
                                           3C   A2  9F 0010B         PUSHAB  60(R2)                                       ;
                                    69          02  FB 0010E 5$:     CALLS   #2, LIB$FREE_VM                              ;
                                           04   AE  9F 00111 6$:     PUSHAB  DCB                                          ; 5418
                              04    AE     70   8F  9A 00114         MOVZBL  #112, 4(SP)                                  ;
                                           04   AE  9F 00119         PUSHAB  4(SP)                                        ;
                                    69          02  FB 0011C         CALLS   #2, LIB$FREE_VM                              ;
                                    50          57  D0 0011F 7$:     MOVL    STATUS, R0                                   ; 5420
                                                    04 00122         RET                                                  ;
                                    56     04   AE  D0 00123 8$:     MOVL    DCB, R6                                      ; 5423
                                    50     02   A6  3C 00127         MOVZWL  2(R6), R0                                    ;
                                                50  D6 0012B         INCL    R0                                           ;
          50           00           6E          00  2C 0012D         MOVC5   #0, (SP), #0, R0, @76(R6)                    ;
                                           4C   B6     00132
                              20    A6     20   A6  9E 00134         MOVAB   32(R6), 32(R6)                               ; 5428
                              24    A6     20   A6  9E 00139         MOVAB   32(R6), 36(R6)                               ; 5429
                                    50     08   A6  9E 0013E         MOVAB   8(R6), DESC                                  ; 5434
                              02    A0   020E   8F  B0 00142         MOVW    #526, 2(DESC)                                ; 5437
```

```
                                 0C    BC           56  D0 00148          MOVL     R6, @NEW_DISPLAY_ID                       ; 5442
                                       50           01  D0 0014C          MOVL     #1, R0                                    ; 5444
                                                        04 0014F          RET                                                ; 5445

; Routine Size:  336 bytes,    Routine Base:  _SMG$CODE + 1B9D


; 5214          5446  1 !<BLF/PAGE>
```

```
1 %SBTTL 'SMG$$CREATE_WCB - Create WCB and its buffers'
1 GLOBAL ROUTINE SMG$$CREATE_WCB (
1                               ROWS,
1                               COLS,
1                               WCB_ADDR
1                                       ) =
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine allocates space for the window control block and
1 !       its window text and attribute buffers and initializes them.
1 !       Two sets of these two buffers are built -- one to reflect what
1 !       is currently on the screen and one to build up what the next
1 !       screen image should look like.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       ret_status.wlc.v = SMG$$CREATE_WCB (    ROWS.rl.r,
1 !                                               COLS.rl.r,
1 !                                               WCB_ADDR.wl.r)
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       ROWS.rl.r       No. of rows in each of the buffers
1 !
1 !       COLS.rl.r       No. of columns in each of the buffers
1 !
1 !       WCB_ADDR.wl.r   Address of the newly-created WCB -- returned to
1 !                       caller.
1 !
1 !
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !       SS$_NORMAL      Normal successful completion
1 !       LIB$_INSVIRMEM  Insufficient virtual memory to allocate needed
1 !                       buffer.
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !--
2   BEGIN
2   LOCAL
2       WCB : REF $WCB_DECL,    ! Address of WCB allocated.
2       STATUS;                 ! Status of subroutine calls
2
2 !+
2 ! Allocate space for the WCB itself.
```

```
!-
    IF NOT (STATUS = LIB$GET_VM (%REF (WCB_K_SIZE), WCB))
    THEN
        RETURN (.STATUS);

    CH$FILL (0, WCB_K_SIZE, .WCB);      ! Clear all fields to default 0

    WCB [WCB_L_BUFSIZE] = ..ROWS * ..COLS; ! Overall size of each buffer

!+
! Attempt to get space for all 4 buffers at once, returning an error if
! we can't.
!-
    IF NOT (STATUS = LIB$GET_VM ( %REF (4 * .WCB [WCB_L_BUFSIZE]),
                                  WCB [WCB_A_TEXT_BUF]))
    THEN
        BEGIN   ! No more space
        !+
        ! If we can't get space for buffers, we might as well give back
        ! the WCB space itself.
        !-
        LIB$FREE_VM (%REF (WCB_K_SIZE), WCB);
        RETURN (.STATUS);
        END;    ! No more space

!+
! Carve up the space gotten into the 4 buffers we need.
!-
    WCB [WCB_A_ATTR_BUF]     = .WCB [WCB_A_TEXT_BUF] +     .WCB [WCB_L_BUFSIZE];
    WCB [WCB_A_SCR_TEXT_BUF] = .WCB [WCB_A_TEXT_BUF] + 2 * .WCB [WCB_L_BUFSIZE];
    WCB [WCB_A_SCR_ATTR_BUF] = .WCB [WCB_A_TEXT_BUF] + 3 * .WCB [WCB_L_BUFSIZE];

!+
! Initialize the working buffers.
!-
    CH$FILL (%C' ', .WCB [WCB_L_BUFSIZE], .WCB [WCB_A_TEXT_BUF]);
    CH$FILL (0,     .WCB [WCB_L_BUFSIZE], .WCB [WCB_A_ATTR_BUF]);

!+
! Initialize the buffers representing what's on the screen to non-
! matchable text as an initial state.  This means the first time
! minimum screen update looks at it it will cause the entire window
! to be repainted.
!-
    CH$FILL (-1, .WCB [WCB_L_BUFSIZE], .WCB [WCB_A_SCR_TEXT_BUF]);
    CH$FILL (0,  .WCB [WCB_L_BUFSIZE], .WCB [WCB_A_SCR_ATTR_BUF]);

!+
! Allocate the line characteristic vectors.  There are two of them --
! one for the text buffer and one for the screen text buffer.  We
! allocate and initialize them together for efficiency.
!-
    IF NOT (STATUS = LIB$GET_VM ( %REF (2 * (..ROWS + 1)),
                                  WCB [WCB_A_LINE_CHAR]))
    THEN
        BEGIN   ! Error path
        !+
; 5330          5561  3             ! Give back all space accumulated on this transaction.
; 5331          5562  3             !-
; 5332          5563  3             LIB$FREE_VM ( %REF (4 * .WCB [WCB_L_BUFSIZE]),
; 5333          5564  3                           WCB [WCB_A_TEXT_BUF]);
; 5334          5565  3
; 5335          5566  3             LIB$FREE_VM ( %REF (WCB_K_SIZE), WCB);
; 5336          5567  2             END;      ! Error path
; 5337          5568  2   !+
; 5338          5569  2   ! Clear both buffer to zero at once
; 5339          5570  2   !-
; 5340          5571  2       CH$FILL (0, 2 * (..ROWS + 1), .WCB [WCB_A_LINE_CHAR]);
; 5341          5572  2
; 5342          5573  2   !+
; 5343          5574  2   ! Use upper half of space just allocated and cleared as the line
; 5344          5575  2   ! characteristics vector for the screen text buffer.
; 5345          5576  2   !-
; 5346          5577  2       WCB [WCB_A_SCR_LINE_CHAR] = .WCB [WCB_A_LINE_CHAR] + ..ROWS + 1;
; 5347          5578  2
; 5348          5579  2   !+
; 5349          5580  2   ! Fill in rest of WCB and return the address of the WCB we've built.
; 5350          5581  2   !-
; 5351          5582  2       WCB [WCB_W_NO_ROWS] = ..ROWS;
; 5352          5583  2       WCB [WCB_W_ROW_START] = 1;
; 5353          5584  2       WCB [WCB_W_NO_COLS] = ..COLS;
; 5354          5585  2       WCB [WCB_W_COL_START]  = 1;
; 5355          5586  2
; 5356          5587  2       .WCB_ADDR = .WCB;
; 5357          5588  2
; 5358          5589  2       RETURN (SS$_NORMAL);
; 5359          5590  1       END;                        ! Routine SMG$$CREATE_WCB
```

```
                                                OFFC 00000          .ENTRY   SMG$$CREATE_WCB, Save R2,R3,R4,R5,R6,R7,R8,-; 5448
                                                                             R9,R10,R11                                     ;
                          5B 00000000G    00  9E 00002          MOVAB    LIB$FREE_VM, R11                               ;
                          5E              08  C2 00009          SUBL2    #8, SP                                         ;
                                  04      AE  9F 0000C          PUSHAB   WCB                                            ; 5505
                     04   AE              34  D0 0000F          MOVL     #52, 4(SP)                                     ;
                                  04      AE  9F 00013          PUSHAB   4(SP)                                          ;
                 00000000G        00      02  FB 00016          CALLS    #2, LIB$GET_VM                                 ;
                          5A              50  D0 0001D          MOVL     R0, STATUS                                     ;
                          3A              5A  E9 00020          BLBC     STATUS, 1$                                     ;
                          56      04      AE  D0 00023          MOVL     WCB, R6                                        ; 5509
      34          00      6E              00  2C 00027          MOVC5    #0, (SP), #0, #52, (R6)                        ;
                                          66     0002C                                                                  ;
                          59      04      BC  D0 0002D          MOVL     @ROWS, R9                                      ; 5511
           28     A6      59      08      BC  C5 00031          MULL3    @COLS, R9, 40(R6)                              ;
                                  08      A6  9F 00037          PUSHAB   8(R6)                                          ; 5518
           04     AE      28      A6      02  78 0003A          ASHL     #2, 40(R6), 4(SP)                              ; 5517
                                  04      AE  9F 00040          PUSHAB   4(SP)                                          ;
                 00000000G        00      02  FB 00043          CALLS    #2, LIB$GET_VM                                 ; 5518
                          5A              50  D0 0004A          MOVL     R0, STATUS                                     ;
                          11              5A  E8 0004D          BLBS     STATUS, 2$                                     ;
                                  04      AE  9F 00050          PUSHAB   WCB                                            ; 5525
```

```
                         04   AE              34  D0 00053          MOVL     #52, 4(SP)                                  :
                                        04    AE  9F 00057          PUSHAB   4(SP)                                       :
                              6B              02  FB 0005A          CALLS    #2, LIB$FREE_VM                             :
                              50              5A  D0 0005D 1$:      MOVL     STATUS, R0                                  : 5526
                                                  04 00060          RET                                                  :
                              56        04    AE  D0 00061 2$:      MOVL     WCB, R6                                     : 5532
                              57        08    A6  9E 00065          MOVAB    8(R6), R7                                   :
                              58        28    A6  9E 00069          MOVAB    40(R6), R8                                  :
           0C   A6            67              68  C1 0006D          ADDL3    (R8), (R7), 12(R6)                          :
                              50              68  D0 00072          MOVL     (R8), R0                                    : 5533
                         14   A6        00 B740   3E 00075          MOVAW    @0(R7)[R0], 20(R6)                          :
                50            68              03  C5 0007B          MULL3    #3, (R8), R0                                : 5534
           18   A6            50              67  C1 0007F          ADDL3    (R7), R0, 24(R6)                            :
  68            20            6E              00  2C 00084          MOVC5    #0, (SP), #32, (R8), @0(R7)                 : 5539
                                        00    B7     00089                                                               :
  68            00            6E              00  2C 0008B          MOVC5    #0, (SP), #0, (R8), @12(R6)                 : 5540
                                        0C    B6     00090                                                               :
  68       FF   8F            6E              00  2C 00092          MOVC5    #0, (SP), #-1, (R8), @20(R6)                : 5548
                                        14    B6     00098                                                               :
  68            00            6E              00  2C 0009A          MOVC5    #0, (SP), #0, (R8), @24(R6)                 : 5549
                                        18    B6     0009F                                                               :
                                        2C    A6  9F 000A1          PUSHAB   44(R6)                                      : 5557
                52            59              01  78 000A4          ASHL     #1, R9, R2                                  : 5556
                              52              02  C0 000A8          ADDL2    #2, R2                                      :
                         04   AE              52  D0 000AB          MOVL     R2, 4(SP)                                   :
                                        04    AE  9F 000AF          PUSHAB   4(SP)                                       :
                   00000000G  00              02  FB 000B2          CALLS    #2, LIB$GET_VM                              : 5557
                              5A              50  D0 000B9          MOVL     R0, STATUS                                  :
                              1A              5A  E8 000BC          BLBS     STATUS, 3$                                  :
                                              57  DD 000BF          PUSHL    R7                                          : 5564
           04   AE            68              02  78 000C1          ASHL     #2, (R8), 4(SP)                             : 5563
                                        04    AE  9F 000C6          PUSHAB   4(SP)                                       :
                              6B              02  FB 000C9          CALLS    #2, LIB$FREE_VM                             : 5564
                                        04    AE  9F 000CC          PUSHAB   WCB                                         : 5566
                         04   AE              34  D0 000CF          MOVL     #52, 4(SP)                                  :
                                        04    AE  9F 000D3          PUSHAB   4(SP)                                       :
                              6B              02  FB 000D6          CALLS    #2, LIB$FREE_VM                             :
                              57        04    AE  D0 000D9 3$:      MOVL     WCB, R7                                     : 5571
  52            00            6E              00  2C 000DD          MOVC5    #0, (SP), #0, R2, @44(R7)                   :
                                        2C    B7     000E2                                                               :
                50            59        2C    A7  C1 000E4          ADDL3    44(R7), R9, R0                              : 5577
                         30   A7        01    A0  9E 000E9          MOVAB    1(R0), 48(R7)                               :
                         02   A7              59  B0 000EE          MOVW     R9, 2(R7)                                   : 5582
                              67              01  B0 000F2          MOVW     #1, (R7)                                    : 5583
                         06   A7        08    BC  B0 000F5          MOVW     @COLS, 6(R7)                                : 5584
                         04   A7              01  B0 000FA          MOVW     #1, 4(R7)                                   : 5585
                         0C   BC              57  D0 000FE          MOVL     R7, @WCB_ADDR                               : 5587
                              50              01  D0 00102          MOVL     #1, R0                                      : 5589
                                                  04 00105          RET                                                  : 5590

; Routine Size:  262 bytes,    Routine Base:  _SMG$CODE + 1CED


; 5360          5591  1 !<BLF/PAGE>
```

```
1 %SBTTL 'SMG$$DEALLOCATE_WCB - Get rid of WCB and its buffers'
1 GLOBAL ROUTINE SMG$$DEALLOCATE_WCB (WCB : REF $WCB_DECL) =
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine deallocates space for the window control block and
1 !       its window text and attribute buffers.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       ret_status.wlc.v = SMG$$CREATE_WCB ( WCB.wl.r)
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       WCB.wl.r        Address of the previously-created WCB.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       contents of WCB
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !       SS$_NORMAL      Normal successful completion
1 !       LIB$_xxx        Errors from LIB$FREE_VM
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !--
2    BEGIN
2    LOCAL
2       RET_STATUS,             ! Status to be returned to caller
2       STATUS;                 ! Status of subroutine calls
2
2 !+
2 ! Attempt to deallocate the space for all 4 buffers (text and attr) at
2 ! once.
2 !-
2    RET_STATUS = LIB$FREE_VM ( %REF(4 * .WCB [WCB_L_BUFSIZE]),
2                               WCB [WCB_A_TEXT_BUF]);
2
2 !+
2 ! Attempt to deallocate the alternate character set buffers if they
2 ! exist.
2 !-
2    IF .WCB [WCB_A_CHAR_SET_BUF] NEQ 0
2    THEN
3       BEGIN   ! Free alt char set buffers
3       ! NOTE: Right now we free them separately.  If it turns out
3       ! they are allocated as a adjacent pair, we can deallocate
3       ! them with a single call.
3
3       STATUS = LIB$FREE_VM ( WCB [WCB_L_BUFSIZE],
; 5419          5649 3                                WCB [WCB_A_CHAR_SET_BUF]);
; 5420          5650 3
; 5421          5651 3              IF NOT .STATUS THEN RET_STATUS = .STATUS ; ! Propagate an error
; 5422          5652 3                                                         ! status
; 5423          5653 3
; 5424          5654 3              STATUS = LIB$FREE_VM ( WCB [WCB_L_BUFSIZE],
; 5425          5655 3                                     WCB [WCB_A_SCR_CHAR_SET_BUF]);
; 5426          5656 3
; 5427          5657 3              IF NOT .STATUS THEN RET_STATUS = .STATUS ; ! Propagate an error
; 5428          5658 3                                                         ! status
; 5429          5659 2              END;    ! Free alt char set buffers
; 5430          5660 2
; 5431          5661 2      !+
; 5432          5662 2      ! Deallocate the line characteristics vectors.  These were allocated
; 5433          5663 2      ! as a pair so can be deallocated as a pair.
; 5434          5664 2      !-
; 5435          5665 2          STATUS = LIB$FREE_VM ( %REF ( 2 * (.WCB [WCB_W_NO_ROWS] + 1)),
; 5436          5666 2                                 WCB [WCB_A_LINE_CHAR]);
; 5437          5667 2
; 5438          5668 2          IF NOT .STATUS THEN RET_STATUS = .STATUS;    ! Propagate an error
; 5439          5669 2                                                       ! status
; 5440          5670 2
; 5441          5671 2      !+
; 5442          5672 2      ! Deallocate the WCB itself.
; 5443          5673 2      !-
; 5444          5674 2          STATUS = LIB$FREE_VM (%REF(WCB_K_SIZE), WCB);
; 5445          5675 2          IF NOT .STATUS THEN RET_STATUS = .STATUS ; ! Propagate an error status
; 5446          5676 2
; 5447          5677 2          RETURN (.RET_STATUS);
; 5448          5678 2
; 5449          5679 1          END;                            ! Routine SMG$$DEALLOCATE_WCB



                                      001C 00000          .ENTRY  SMG$$DEALLOCATE_WCB, Save R2,R3,R4              ; 5593
                  54 00000000G    00   9E 00002          MOVAB   LIB$FREE_VM, R4                                  ;
                  5E              04   C2 00009          SUBL2   #4, SP                                           ;
                  52        04    AC   D0 0000C          MOVL    WCB, R2                                          ; 5635
                            08    A2   9F 00010          PUSHAB  8(R2)                                            ;
   04    AE        28   A2        02   78 00013          ASHL    #2, 40(R2), 4(SP)                                ; 5634
                            04    AE   9F 00019          PUSHAB  4(SP)                                            ;
                  64              02   FB 0001C          CALLS   #2, LIB$FREE_VM                                  ; 5635
                  53              50   D0 0001F          MOVL    R0, RET_STATUS                                   ;
                            10    A2   D5 00022          TSTL    16(R2)                                           ; 5641
                                  1E   13 00025          BEQL    2$                                               ;
                            10    A2   9F 00027          PUSHAB  16(R2)                                           ; 5649
                            28    A2   9F 0002A          PUSHAB  40(R2)                                           ; 5648
                  64              02   FB 0002D          CALLS   #2, LIB$FREE_VM                                  ; 5649
                  03              50   E8 00030          BLBS    STATUS, 1$                                       ; 5651
                  53              50   D0 00033          MOVL    STATUS, RET_STATUS                               ;
                            1C    A2   9F 00036  1$:     PUSHAB  28(R2)                                           ; 5655
                            28    A2   9F 00039          PUSHAB  40(R2)                                           ; 5654
                  64              02   FB 0003C          CALLS   #2, LIB$FREE_VM                                  ; 5655
                  03              50   E8 0003F          BLBS    STATUS, 2$                                       ; 5657
                  53              50   D0 00042          MOVL    STATUS, RET_STATUS                               ;
```

SMG
1-0

```
                                                      L  7
SMG$DISPLAY_LIN SMG$DISPLAY_LINKS - Virtual Display Linkages    16-Sep-1984 00:29:22    VAX-11 Bliss-32 V4.0-742          Page 157
1-096           SMG$$DEALLOCATE_WCB - Get rid of WCB and its bu 14-Sep-1984 13:09:43    [SMGRTL.SRC]SMGDISLIN.B32;1           (32)

                                                          2C    A2  9F 00045 2$:      PUSHAB  44(R2)                              ; 5666
                                               51         02    A2  3C 00048          MOVZWL  2(R2), R1                           ; 5665
                         04   AE               51               01  78 0004C          ASHL    #1, R1, 4(SP)                       ;
                                         04    AE               02  C0 00051          ADDL2   #2, 4(SP)                           ;
                                                          04    AE  9F 00055          PUSHAB  4(SP)                               ;
                                               64               02  FB 00058          CALLS   #2, LIB$FREE_VM                     ; 5666
                                               03               50  E8 0005B          BLBS    STATUS, 3$                          ; 5668
                                               53               50  D0 0005E          MOVL    STATUS, RET_STATUS                  ;
                                                          04    AC  9F 00061 3$:      PUSHAB  WCB                                 ; 5674
                                         04    AE               34  D0 00064          MOVL    #52, 4(SP)                          ;
                                                          04    AE  9F 00068          PUSHAB  4(SP)                               ;
                                               64               02  FB 0006B          CALLS   #2, LIB$FREE_VM                     ;
                                               03               50  E8 0006E          BLBS    STATUS, 4$                          ; 5675
                                               53               50  D0 00071          MOVL    STATUS, RET_STATUS                  ;
                                               50               53  D0 00074 4$:      MOVL    RET_STATUS, R0                      ; 5677
                                                                    04 00077          RET                                         ; 5679

; Routine Size:  120 bytes,    Routine Base:  _SMG$CODE + 1DF3


; 5450         5680  1 !<BLF/PAGE>
```

```
1 %SBTTL 'SMG$$DUPL_VIRTUAL_DISPLAY - Duplicate a virtual display'
1 GLOBAL ROUTINE SMG$$DUPL_VIRTUAL_DISPLAY (
1                                                 CURR_DISPLAY_ID,
1                                                 NEW_DISPLAY_ID
1                                               ) =
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine makes a copy of an existing virtual display,
1 !       assigning it a new virtual display number.  The new virtual
1 !       will not be pasted anywhere -- even if the virtual display from
1 !       which it was created was.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       ret_status.wlc.v = SMG$$DUPL_VIRTUAL_DISPLAY (CURR_DISPLAY_ID,
1 !                                                     NEW_DISPLAY_ID)
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       CURR_DISPLAY_ID.rl.r    Display id of virtual display to be
1 !                               replicated.
1 !
1 !       NEW_DISPLAY_ID.wl.r     Display id of newly-created virtual
1 !                               display.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !       SS$_NORMAL      Normal successful completion
1 !       LIB$_INSVIRMEM  Insufficient virtual memory to allocate needed
1 !                       buffer.
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !--
2     BEGIN
2     LOCAL
2         DCB     : REF $DCB_DECL,        ! Address of current DCB.
2         DCB_NEW : REF $DCB_DECL,        ! Address of new DCB
2         STATUS;         ! Status of subroutine calls
2
2     $SMG$GET_DCB (.CURR_DISPLAY_ID, DCB); ! Get addr of DCB for current
2                                           ! display
2
2 !+
2 ! If a backup DCB does not yet exist, allocate one.
2 ! Make a new virtual display using the sizes and attributes of the old
2 ! one.  Quit if we can't.
```

```
2 !-
2     IF .DCB [DCB_A_BACKUP_DCB] EQL 0
2     THEN
3         BEGIN   ! 1st time, create the backup
4         IF NOT (STATUS = SMG$CREATE_VIRTUAL_DISPLAY (
4                         %REF (.DCB [DCB_W_NO_ROWS]),           ! #rows
4                         %REF (.DCB [DCB_W_NO_COLS]),           ! #cols
4                         .NEW_DISPLAY_ID,                       ! new id
4                         %REF (.DCB [DCB_B_DEF_DISPLAY_ATTR]),  ! disp
4                         %REF (.DCB [DCB_B_DEF_VIDEO_ATTR]),    ! video
4                         %REF (.DCB [DCB_B_DEF_CHAR_SET])))     ! alt char set
3         THEN
3             RETURN (.STATUS);
3
3         $SMG$GET_DCB (.NEW_DISPLAY_ID, DCB_NEW); ! Get DCB address of new
3
3         !+
3         ! Store the new display id in the new DCB.
3         !-
3         DCB_NEW [DCB_L_DID] = ..NEW_DISPLAY_ID;
3
3         END     ! 1st time, create the backup
2
2     ELSE
3         BEGIN   ! Backup already exists
3         .NEW_DISPLAY_ID = .DCB [DCB_A_BACKUP_DCB]; ! Return id of existing
3         DCB_NEW =         .DCB [DCB_A_BACKUP_DCB];
2         END;    ! Backup already exists
2
2 !+
2 ! Now need to copy over the current text and attribute buffers from
2 ! the current to the new.
2 !-
2
2     CH$MOVE ( .DCB [DCB_L_BUFSIZE],                                    ! #bytes
2               .DCB [DCB_A_TEXT_BUF],                                   ! from
2               .DCB_NEW [DCB_A_TEXT_BUF]);                              ! to
2
2     CH$MOVE ( .DCB [DCB_L_BUFSIZE],                                    ! #bytes
2               .DCB [DCB_A_ATTR_BUF],                                   ! from
2               .DCB_NEW [DCB_A_ATTR_BUF]);                              ! to
2
2 !+
2 ! Copy over the line characteristics vector.
2 !-
2     CH$MOVE ( .DCB [DCB_W_NO_ROWS] + 1,
2               .DCB [DCB_A_LINE_CHAR],
2               .DCB_NEW [DCB_A_LINE_CHAR]);
2
2 !+
2 ! Copy over stuff relating to borders and labels.
2 !-
2     IF .DCB_NEW [DCB_V_BORDERED]
2     THEN
3         BEGIN   ! Bordered
3         LOCAL
3             DESC : REF BLOCK [8,BYTE];  ! Pointer to dynamic string
```

```
3                                                 ! descriptor in DCB for border
3                                                 ! label
3
3           DESC = DCB [DCB_Q_LABEL_DESC];
3
3           !+
3           ! If label exists, make a copy.
3           !-
3           IF .DESC [DSC$A_POINTER] NEQ 0
3           THEN
4               BEGIN       ! Labeled
5               IF NOT (STATUS = LIB$SCOPY_DXDX ( .DESC,
5                                                 DCB_NEW [DCB_Q_LABEL_DESC] ))
4               THEN
4                   RETURN (.STATUS);
4
4               DCB_NEW [DCB_W_LABEL_UNITS]    = .DCB [DCB_W_LABEL_UNITS];
4               DCB_NEW [DCB_B_LABEL_POS]      = .DCB [DCB_B_LABEL_POS];
4               DCB_NEW [DCB_B_LABEL_CHAR_SET] = .DCB [DCB_B_LABEL_CHAR_SET];
4               DCB_NEW [DCB_V_LABEL_CENTER]   = .DCB [DCB_V_LABEL_CENTER];
4
3               END;        ! Labeled
2           END;    ! Bordered
2
2   !+
2   ! If alternate character set buffer involved, copy it over as well.
2   !-
2       IF .DCB_NEW [DCB_A_CHAR_SET_BUF] NEQ 0
2       THEN
3           BEGIN   ! Alt char set buffer involved
4           IF NOT (STATUS = LIB$GET_VM (DCB [DCB_L_BUFSIZE],
4                                       DCB_NEW [DCB_A_CHAR_SET_BUF]))
3           THEN
3               RETURN (.STATUS);
3
3           CH$MOVE (.DCB [DCB_L_BUFSIZE],                          ! Num.
3                   .DCB [DCB_A_CHAR_SET_BUF],                      ! From
3                   .DCB_NEW [DCB_A_CHAR_SET_BUF]);                 ! To
3
2           END;    ! Alt char set buffer involved
2
2   !+
2   ! Also preserve the current cursor postion.
2   !-
2       DCB_NEW [DCB_W_CURSOR_ROW] = .DCB [DCB_W_CURSOR_ROW];
2       DCB_NEW [DCB_W_CURSOR_COL] = .DCB [DCB_W_CURSOR_COL];
2
2       RETURN (SS$_NORMAL);
1       END;                        ! Routine SMG$$DUPL_VIRTUAL_DISPLAY


                                03FC 00000        .ENTRY  SMG$$DUPL_VIRTUAL_DISPLAY, Save R2,R3,R4,-  ; 5682
                                                          R5,R6,R7,R8,R9                              ;
                 59 00000000G  8F   D0 00002        MOVL    #SMG$_INVDIS_ID, R9                         ;
```

```
                                                                   C  B
SMG$DISPLAY_LIN SMG$DISPLAY_LINKS - Virtual Display Linkages     16-Sep-1984 00:29:22    VAX-11 Bliss-32 V4.0-742             Page 161
1-096           SMG$$DUPL_VIRTUAL_DISPLAY - Duplicate a virtual 14-Sep-1984 13:09:43    [SMGRTL.SRC]SMGDISLIN.B32;1               (33)

                                    5E           14  C2 00009       SUBL2   #20, SP                                   ;
                                    50      04   BC  D0 0000C       MOVL    @CURR_DISPLAY_ID, R0                      ; 5731
                                04  BC      38   A0  D1 00010       CMPL    56(R0), @CURR_DISPLAY_ID                  ;
                                                 06  12 00015       BNEQ    1$                                        ;
                                    11      44   A0  91 00017       CMPB    68(R0), #17                               ;
                                                 04  13 0001B       BEQL    2$                                        ;
                                    50           59  D0 0001D 1$:   MOVL    R9, R0                                    ;
                                                     04 00020       RET                                               ;
                                    56      04   BC  D0 00021 2$:   MOVL    @CURR_DISPLAY_ID, DCB                     ;
                                    50      40   A6  D0 00025       MOVL    64(DCB), R0                               ; 5739
                                                 56  12 00029       BNEQ    5$                                        ;
                                10  AE      30   A6  9A 0002B       MOVZBL  48(DCB), 16(SP)                           ; 5748
                                            10   AE  9F 00030       PUSHAB  16(SP)                                    ;
                                10  AE      2E   A6  9A 00033       MOVZBL  46(DCB), 16(SP)                           ; 5747
                                            10   AE  9F 00038       PUSHAB  16(SP)                                    ;
                                10  AE      2F   A6  9A 0003B       MOVZBL  47(DCB), 16(SP)                           ; 5746
                                            10   AE  9F 00040       PUSHAB  16(SP)                                    ;
                                            08   AC  DD 00043       PUSHL   NEW_DISPLAY_ID                            ; 5745
                                14  AE      06   A6  3C 00046       MOVZWL  6(DCB), 20(SP)                            ; 5744
                                            14   AE  9F 0004B       PUSHAB  20(SP)                                    ;
                                14  AE      02   A6  3C 0004E       MOVZWL  2(DCB), 20(SP)                            ; 5743
                                            14   AE  9F 00053       PUSHAB  20(SP)                                    ;
                              FCD7  CF           06  FB 00056       CALLS   #6, SMG$$CREATE_VIRTUAL_DISPLAY           ;
                                    58           50  D0 0005B       MOVL    R0, STATUS                                ;
                                    5D           58  E9 0005E       BLBC    STATUS, 7$                                ; 5742
                                    50      08   BC  D0 00061       MOVL    @NEW_DISPLAY_ID, R0                       ; 5752
                                08  BC      38   A0  D1 00065       CMPL    56(R0), @NEW_DISPLAY_ID                   ;
                                                 06  12 0006A       BNEQ    3$                                        ;
                                    11      44   A0  91 0006C       CMPB    68(R0), #17                               ;
                                                 04  13 00070       BEQL    4$                                        ;
                                    50           59  D0 00072 3$:   MOVL    R9, R0                                    ;
                                                     04 00075       RET                                               ;
                                    57      08   BC  D0 00076 4$:   MOVL    @NEW_DISPLAY_ID, DCB_NEW                  ;
                                38  A7      08   BC  D0 0007A       MOVL    @NEW_DISPLAY_ID, 56(DCB_NEW)              ; 5757
                                                 07  11 0007F       BRB     6$                                        ; 5739
                                08  BC           50  D0 00081 5$:   MOVL    R0, @NEW_DISPLAY_ID                       ; 5763
                                    57           50  D0 00085       MOVL    R0, DCB_NEW                               ; 5764
              10    B7          10  B6      3C   A6  28 00088 6$:   MOVC3   60(DCB), @16(DCB), @16(DCB_NEW)           ; 5774
              14    B7          14  B6      3C   A6  28 0008F       MOVC3   60(DCB), @20(DCB), @20(DCB_NEW)           ; 5778
                                    50      02   A6  3C 00096       MOVZWL  2(DCB), R0                                ; 5783
                                                 50  D6 0009A       INCL    R0                                        ;
              4C    B7          4C  B6           50  28 0009C       MOVC3   R0, @76(DCB), @76(DCB_NEW)                ; 5785
                                    31      2F   A7  E9 000A2       BLBC    47(DCB_NEW), 8$                           ; 5790
                                    50      08   A6  9E 000A6       MOVAB   8(R6), DESC                               ; 5798
                                            04   A0  D5 000AA       TSTL    4(DESC)                                   ; 5803
                                                 28  13 000AD       BEQL    8$                                        ;
                                            08   A7  9F 000AF       PUSHAB  8(DCB_NEW)                                ; 5807
                                                 50  DD 000B2       PUSHL   DESC                                      ;
                         00000000G  00           02  FB 000B4       CALLS   #2, LIB$SCOPY_DXDX                        ;
                                    58           50  D0 000BB       MOVL    R0, STATUS                                ;
                                    2E           58  E9 000BE 7$:   BLBC    STATUS, 9$                                ;
                                2C  A7      2C   A6  B0 000C1       MOVW    44(DCB), 44(DCB_NEW)                      ; 5811
                                31  A7      31   A6  B0 000C6       MOVW    49(DCB), 49(DCB_NEW)                      ; 5812
        50    34    A6              01           02  EF 000CB       EXTZV   #2, #1, 52(DCB), R0                       ; 5814
  34    A7          01              02           50  F0 000D1       INSV    R0, #2, #1, 52(DCB_NEW)                   ;
                                            18   A7  D5 000D7 8$:   TSTL    24(DCB_NEW)                               ; 5822
                                                 1E  13 000DA       BEQL    11$                                       ;
```

```
                                       18  A7  9F 000DC          PUSHAB  24(DCB_NEW)                                  ; 5826
                                       3C  A6  9F 000DF          PUSHAB  60(DCB)                                      ; 5825
                     00000000G  00         02  FB 000E2          CALLS   #2, LIB$GET_VM                               ; 5826
                                58         50  D0 000E9          MOVL    R0, STATUS                                   ;
                                04         58  E8 000EC          BLBS    STATUS, 10$                                  ;
                                50         58  D0 000EF 9$:      MOVL    STATUS, R0                                   ; 5828
                                               04 000F2          RET                                                  ;
               18   B7      18  B6     3C  A6  28 000F3 10$:     MOVC3   60(DCB), @24(DCB), @24(DCB_NEW)              ; 5832
                            28  A7     28  A6  D0 000FA 11$:     MOVL    40(DCB), 40(DCB_NEW)                         ; 5839
                                50         01  D0 000FF          MOVL    #1, R0                                       ; 5842
                                               04 00102          RET                                                  ; 5843

; Routine Size:  259 bytes,    Routine Base:  _SMG$CODE + 1E6B


; 5615          5844  1 !<BLF/PAGE>
```

```
1 %SBTTL 'SMG$$LOCATE_PP - Locate Pasting packet for given display and pasteboard'
1 GLOBAL ROUTINE SMG$$LOCATE_PP ( DCB  : REF $DCB_DECL,
1                                 PBCB : REF $PBCB_DECL,
1                                 PP ) =
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Locate the address of the pasting packet that joins this
1 !       virtual display to this pasteboard.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       ret_status.wlc.v = SMG$$LOCATE_PP (     DCB.rab.r,
1 !                                               PBCB.rab.r,
1 !                                               PP.wl.r)
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       DCB.rab.r       Address of a virtual display control block.
1 !
1 !       PBCB.rab.r      Address of a pasteboard control block.
1 !
1 !       PP.wl.r         Return address of the pasting packet that
1 !                       represents the pasting of the given virtual
1 !                       display to the given pasteboard control block.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       None
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       None
1 !
1 ! COMPLETION STATUS:
1 !
1 !       SS$_NORMAL      Normal successful completion
1 !       SMG$_NOTPASTED  Given display is not pasted to given pasteboard
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !--
2     BEGIN
2     LOCAL
2       SEARCH_DCB : REF $DCB_DECL,     ! Addr of the DCB we'll actually
2                                       ! search for
2
2       CURR_PP : REF $PP_DECL;         ! Addr of pasting packet being
2                                       ! inspected.
2
2     CURR_PP = .DCB [DCB_A_PP_NEXT];   ! Start with 1st PP in chain
2
2 !+
2 ! If the virtual display is currently batched, the batch level will be non-zero.
2 ! This means a match needs to be found on the backup DCB address instead of the
2 ! DCB address.
```

```
; 5674          5902  2 !-
; 5675          5903  2     SEARCH_DCB = .DCB;
; 5676          5904  2
; 5677          5905  2     IF .DCB [DCB_L_BATCH_LEVEL] NEQ 0           ! Currently batched
; 5678          5906  2     THEN
; 5679          5907  2         SEARCH_DCB = .DCB [DCB_A_BACKUP_DCB];
; 5680          5908  2
; 5681          5909  2
; 5682          5910  2     WHILE .CURR_PP NEQ DCB [DCB_A_PP_NEXT]      ! While any remain
; 5683          5911  2     DO
; 5684          5912  3         BEGIN   ! Search for packet with matching PBCB addr
; 5685          5913  3         IF .CURR_PP [PP_A_DCB_ADDR]  EQL .SEARCH_DCB    AND
; 5686          5914  3            .CURR_PP [PP_A_PBCB_ADDR] EQL .PBCB
; 5687          5915  3         THEN
; 5688          5916  4             BEGIN       ! Desired packet found
; 5689          5917  4             .PP = .CURR_PP;
; 5690          5918  4             RETURN (SS$_NORMAL);        ! Return success
; 5691          5919  3             END;        ! Desired packet found
; 5692          5920  3
; 5693          5921  3         CURR_PP = .CURR_PP [PP_A_NEXT_DCB]; ! Otherwise step along DCB
; 5694          5922  3                                             ! side of chain
; 5695          5923  2         END;    ! Search for packet with matching PBCB addr
; 5696          5924  2
; 5697          5925  2 !+
; 5698          5926  2 ! If we fall out of the while loop, this virtual display is not pasted
; 5699          5927  2 ! to the specified pasteboard -- according to the pasting packets.
; 5700          5928  2 !-
; 5701          5929  2     .PP = 0;    ! To reduce liklihood someone will try to use it
; 5702          5930  2                 ! and disregard status.
; 5703          5931  2     RETURN (SMG$_NOTPASTED);        ! Return failure
; 5704          5932  1     END;                            ! Routine SMG$$LOCATE_PP




                                   000C 00000         .ENTRY  SMG$$LOCATE_PP, Save R2,R3                             ; 5846
              50        04    AC   D0 00002           MOVL    DCB, R0                                                ; 5896
              51        20    A0   D0 00006           MOVL    32(R0), CURR_PP                                        ;
              53              50   D0 0000A           MOVL    R0, SEARCH_DCB                                         ; 5903
                        1C    A0   D5 0000D           TSTL    28(R0)                                                 ; 5905
                              04   13 00010           BEQL    1$                                                     ;
              53        40    A0   D0 00012           MOVL    64(R0), SEARCH_DCB                                     ; 5907
              52        20    A0   9E 00016   1$:     MOVAB   32(R0), R2                                             ; 5910
              52              51   D1 0001A           CMPL    CURR_PP, R2                                            ;
                              1A   13 0001D           BEQL    3$                                                     ;
              53        10    A1   D1 0001F           CMPL    16(CURR_PP), SEARCH_DCB                                ; 5913
                              0F   12 00023           BNEQ    2$                                                     ;
        08    AC        14    A1   D1 00025           CMPL    20(CURR_PP), PBCB                                      ; 5914
                              08   12 0002A           BNEQ    2$                                                     ;
        0C    BC              51   D0 0002C           MOVL    CURR_PP, @PP                                           ; 5917
              50              01   D0 00030           MOVL    #1, R0                                                 ; 5918
                                   04 00033           RET                                                            ;
              51              61   D0 00034   2$:     MOVL    (CURR_PP), CURR_PP                                     ; 5921
                              DD   11 00037           BRB     1$                                                     ; 5910
                        0C    BC   D4 00039   3$:     CLRL    @PP                                                    ; 5929
              50 00000000G    8F   D0 0003C           MOVL    #SMG$_NOTPASTED, R0                                    ; 5931
```

```
                                                   04 00043          RET                                          ; 5932

; Routine Size:  68 bytes,    Routine Base:  _SMG$CODE + 1F6E

; 5705          5933  1 !<BLF/PAGE>
```

```
1 %SBTTL 'SMG$$PASTE_VIRTUAL_DISPLAY - Paste virtual display to pasteboard'
1 GLOBAL ROUTINE SMG$$PASTE_VIRTUAL_DISPLAY (
1                                               DCB : REF $DCB_DECL,
1                                               PBCB : REF $PBCB_DECL,
1                                               PASTEBOARD_ROW,
1                                               PASTEBOARD_COL
1                                             ) =
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       The specified virtual display is "pasted" (oriented
1 !       with respect to) a pasteboard.  This makes the display visible.
1 !       This is the inner paste routine.  It assumes input parameters
1 !       are all present and valid.  Further assumes that display
1 !       specified by DCB is not already pasted to pasteboard specified
1 !       by PBCB.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       ret_status.wlc.v = SMG$$PASTE_VIRTUAL_DISPLAY (
1 !                                                   DCB.rab.r,
1 !                                                   PBCB.rab.r,
1 !                                                   PASTEBOARD_ROW.rl.r,
1 !                                                   PASTEBOARD_COL.rl.r)
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       DCB.rab.r               Address of virtual display to be pasted.
1 !
1 !       PBCB.rab.r              Address of the pasteboard on
1 !                               which the pasting is to take place.
1 !
1 !       PASTEBOARD_ROW.rl.r     Row on pasteboard which is to contain
1 !                               row 1 of the specified virtual display.
1 !
1 !       PASTEBOARD_COL.rl.r     Column on pasteboard which is to contain
1 !                               column 1 of the specified virtual
1 !                               display.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       None
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       None
1 !
1 ! COMPLETION STATUS:
1 !
1 !       SS$_NORMAL      Normal successful completion
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !--
2     BEGIN
2     LOCAL
```

```
            STATUS,                         ! Status of subroutine calls

            PP      : REF $PP_DECL,         ! Addr of the pasting packet
                                            ! being created.

            WCB     : REF $WCB_DECL;        ! Addr. of window control block

    !+
    ! Get space for pasting packet.
    !-
        IF NOT (STATUS = LIB$GET_VM ( %REF (PP_K_SIZE), PP))
        THEN
            RETURN (.STATUS);

        CH$FILL (0, PP_K_SIZE, .PP);        ! Clear all fields to default 0

    !+
    ! Initialize pasting packet
    !-
        PP [PP_A_DCB_ADDR]  = .DCB;
        PP [PP_A_PBCB_ADDR] = .PBCB;
        PP [PP_W_ROW]       = ..PASTEBOARD_ROW;
        PP [PP_W_COL]       = ..PASTEBOARD_COL;

    !+
    ! Plug this packet onto both queues.
    !-
        $SMG$INSERT_AT_HEAD ( PP [PP_A_NEXT_DCB],  DCB [DCB_A_PP_NEXT]);
        $SMG$INSERT_AT_HEAD ( PP [PP_A_NEXT_PBCB], PBCB [PBCB_A_PP_NEXT]);

    !+
    ! If the display is batched, we want the backpointer in the PP to be
    ! pointing to our backup DCB.
    !-
        IF .DCB [DCB_L_BATCH_LEVEL] NEQ 0
        THEN
            PP [PP_A_DCB_ADDR] = .DCB [DCB_A_BACKUP_DCB];

    !+
    ! Recalc. occlusions introduced by this new pasting.
    !-
        IF NOT ( STATUS = SMG$$CHECK_OCCLUSION_FIRST ( .PBCB))
        THEN
            RETURN (.STATUS);

    !+
    ! Calculate the transformation constants needed to copy this display's
    ! buffers into the associated window's buffers.
    !-
        IF NOT ( STATUS = SMG$$CALC_PASTE_TRANSF (.PP))
        THEN
            RETURN (.STATUS);

    !+
    ! If pasteboard batching enabled, quit here.
    !-
        IF .PBCB [PBCB_L_BATCH_LEVEL] NEQ 0
```

```
; 5821      6048 2      THEN
; 5822      6049 2          RETURN ( SMG$_BATWAS_ON);
; 5823      6050 2
; 5824      6051 2  !+
; 5825      6052 2  ! Force physical display's cursor to be where this virtual display's
; 5826      6053 2  ! cursor is.
; 5827      6054 2  ! Chose between current DCB and backup DCB.
; 5828      6055 2  !-
; 5829      6056 2      WCB = .PBCB [PBCB_A_WCB];
; 5830      6057 2
; 5831      6058 2      IF .DCB [DCB_L_BATCH_LEVEL] EQL 0
; 5832      6059 2      THEN
; 5833      6060 3          BEGIN   ! Get from current DCB
; 5834      6061 3          WCB [WCB_W_CURR_CUR_ROW] = .DCB [DCB_W_CURSOR_ROW] - 1 + .PP [PP_W_ROW];
; 5835      6062 3          WCB [WCB_W_CURR_CUR_COL] = .DCB [DCB_W_CURSOR_COL] - 1 + .PP [PP_W_COL];
; 5836      6063 3          END     ! Get from current DCB
; 5837      6064 3
; 5838      6065 2      ELSE
; 5839      6066 2
; 5840      6067 3          BEGIN   ! Get from backup DCB
; 5841      6068 3          LOCAL
; 5842      6069 3              BACK_DCB : REF $DCB_DECL; ! Addr of backup DCB
; 5843      6070 3
; 5844      6071 3          BACK_DCB = .DCB [DCB_A_BACKUP_DCB];
; 5845      6072 3          WCB [WCB_W_CURR_CUR_ROW] = .BACK_DCB [DCB_W_CURSOR_ROW] - 1 + .PP [PP_W_ROW];
; 5846      6073 3          WCB [WCB_W_CURR_CUR_COL] = .BACK_DCB [DCB_W_CURSOR_COL] - 1 + .PP [PP_W_COL];
; 5847      6074 2          END;    ! Get from backup DCB
; 5848      6075 2
; 5849      6076 2  !+
; 5850      6077 2  ! Move stuff from virtual display to pasteboard buffer and caused it
; 5851      6078 2  ! to be output if pasteboard is not batched.
; 5852      6079 2  !-
; 5853      6080 2      IF .PBCB [PBCB_L_BATCH_LEVEL] EQL 0
; 5854      6081 2      THEN
; 5855      6082 3          BEGIN
; 5856      6083 3          PBCB [PBCB_W_FIRST_CHANGED_ROW] = 1;
; 5857      6084 3          PBCB [PBCB_W_LAST_CHANGED_ROW]  = .PBCB [PBCB_B_ROWS];
; 5858      6085 3          PBCB [PBCB_W_FIRST_CHANGED_COL] = 1;
; 5859      6086 3          PBCB [PBCB_W_LAST_CHANGED_COL]  = .PBCB [PBCB_W_WIDTH];
; 5860      6087 3          RETURN (SMG$$FILL_WINDOW_BUFFER (.PP));
; 5861      6088 2          END;
; 5862      6089 2
; 5863      6090 2  !+
; 5864      6091 2  ! Else just return Batch-was-On status.
; 5865      6092 2  !-
; 5866      6093 2      RETURN ( SMG$_BATWAS_ON);
; 5867      6094 2
; 5868      6095 1      END;                        ! Routine SMG$$PASTE_VIRTUAL_DISPLAY



                                 00FC 00000          .ENTRY  SMG$$PASTE_VIRTUAL_DISPLAY, Save R2,R3,R4,-  ; 5935
                                                             R5,R6,R7                                     ;
               5E            08  C2 00002          SUBL2   #8, SP                                       ;
                     04      AE  9F 00005          PUSHAB  PP                                           ; 6001
```

```
                                                           K  8
SMG$DISPLAY_LIN SMG$DISPLAY_LINKS - Virtual Display Linkages     16-Sep-1984 00:29:22    VAX-11 Bliss-32 V4.0-742                    Page 169
1-096           SMG$$PASTE_VIRTUAL_DISPLAY - Paste virtual disp 14-Sep-1984 13:09:43    [SMGRTL.SRC]SMGDISLIN.B32;1                     (35)

                                 04  AE          37  D0 00008        MOVL    #55, 4(SP)
                                              04 AE  9F 0000C        PUSHAB  4(SP)
                          00000000G  00          02  FB 0000F        CALLS   #2, LIB$GET_VM
                                     57          50  D0 00016        MOVL    R0, STATUS
                                     5B          57  E9 00019        BLBC    STATUS, 2$
                                     56       04 AE  D0 0001C        MOVL    PP, R6                                     6005
           37           00           6E          00  2C 00020        MOVC5   #0, (SP), #0, #55, (R6)
                                                 66     00025
                                 10  A6       04 AC  7D 00026        MOVQ    DCB, 16(R6)                                6010
                                 18  A6       0C BC  B0 0002B        MOVW    @PASTEBOARD_ROW, 24(R6)                    6012
                                 1A  A6       10 BC  B0 00030        MOVW    @PASTEBOARD_COL, 26(R6)                    6013
                        50       04  AC          20  C1 00035        ADDL3   #32, DCB, R0                               6018
                                     60          66  0E 0003A        INSQUE  (R6), (R0)
                                     50       04 AE  D0 0003D        MOVL    PP, R0                                     6019
                                 08  BC       08 A0  0E 00041        INSQUE  8(R0), @PBCB
                                     52       04 AC  D0 00046        MOVL    DCB, R2                                    6025
                                              1C A2  D5 0004A        TSTL    28(R2)
                                                 09  13 0004D        BEQL    1$
                                     50       04 AE  D0 0004F        MOVL    PP, R0                                     6027
                                 10  A0       40 A2  D0 00053        MOVL    64(R2), 16(R0)
                                     53       08 AC  D0 00058 1$:    MOVL    PBCB, R3                                   6032
                                                 53  DD 0005C        PUSHL   R3
                               FA33  CF          01  FB 0005E        CALLS   #1, SMG$$CHECK_OCCLUSION_FIRST
                                     57          50  D0 00063        MOVL    R0, STATUS
                                     0E          57  E9 00066        BLBC    STATUS, 2$
                                              04 AE  DD 00069        PUSHL   PP                                         6040
                               F649  CF          01  FB 0006C        CALLS   #1, SMG$$CALC_PASTE_TRANSF
                                     57          50  D0 00071        MOVL    R0, STATUS
                                     04          57  E8 00074        BLBS    STATUS, 3$
                                     50          57  D0 00077 2$:    MOVL    STATUS, R0                                 6042
                                                     04 0007A        RET
                                     56     00A4 C3  D0 0007B 3$:    MOVL    164(R3), R6                                6047
                                                 7C  12 00080        BNEQ    6$
                                     51       08 A3  D0 00082        MOVL    8(R3), WCB                                 6056
                                     55       04 AE  D0 00086        MOVL    PP, R5                                     6061
                                     54       04 AE  D0 0008A        MOVL    PP, R4                                     6062
                                              1C A2  D5 0008E        TSTL    28(R2)                                     6058
                                                 22  12 00091        BNEQ    4$
                                     50       28 A2  3C 00093        MOVZWL  40(R2), R0                                 6061
                                     57       18 A5  32 00097        CVTWL   24(R5), R7
                                     50          57  C0 0009B        ADDL2   R7, R0
                   20   A1           50          01  A3 0009E        SUBW3   #1, R0, 32(WCB)
                                     52       2A A2  3C 000A3        MOVZWL  42(R2), R2                                 6062
                                     50       1A A4  32 000A7        CVTWL   26(R4), R0
                                     52          50  C0 000AB        ADDL2   R0, R2
                   22   A1           52          01  A3 000AE        SUBW3   #1, R2, 34(WCB)
                                                 24  11 000B3        BRB     5$                                         6058
                                     50       40 A2  D0 000B5 4$:    MOVL    64(R2), BACK_DCB                           6071
                                     52       28 A0  3C 000B9        MOVZWL  40(BACK_DCB), R2                           6072
                                     57       18 A5  32 000BD        CVTWL   24(R5), R7
                                     52          57  C0 000C1        ADDL2   R7, R2
                   20   A1           52          01  A3 000C4        SUBW3   #1, R2, 32(WCB)
                                     50       2A A0  3C 000C9        MOVZWL  42(BACK_DCB), R0                           6073
                                     52       1A A4  32 000CD        CVTWL   26(R4), R2
                                     50          52  C0 000D1        ADDL2   R2, R0
                   22   A1           50          01  A3 000D4        SUBW3   #1, R0, 34(WCB)
                                                 56  D5 000D9 5$:    TSTL    R6                                         6080
```

```
                                       21  12 000DB           BNEQ    6$                                 ;
                  00A8   C3            01  B0 000DD           MOVW    #1, 168(R3)                        ; 6083
                  00AA   C3       5F   A3  9B 000E2           MOVZBW  95(R3), 170(R3)                    ; 6084
                  00AC   C3            01  B0 000E8           MOVW    #1, 172(R3)                        ; 6085
                  00AE   C3       5A   A3  B0 000ED           MOVW    90(R3), 174(R3)                    ; 6086
                                  04   AE  DD 000F3           PUSHL   PP                                 ; 6087
             00000000G   00            01  FB 000F6           CALLS   #1, SMG$$FILL_WINDOW_BUFFER        ;
                                           04 000FD           RET                                        ;
                         50 00000000G  8F  D0 000FE 6$:       MOVL    #SMG$_BATWAS_ON, R0                ; 6093
                                           04 00105           RET                                        ; 6095

; Routine Size:  262 bytes,    Routine Base:  _SMG$CODE + 1FB2

; 5869          6096  1 !<BLF/PAGE>
```

```
1 %SBTTL 'SMG$$RECALC_PP_FIELDS - Recalc. Pasting Packet fields'
1 GLOBAL ROUTINE SMG$$RECALC_PP_FIELDS (
1                                     DCB  : REF $DCB_DECL
1                                   )=
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine recalculates fields in the pasting packet that
1 !       need to change.
1 !       It walks the chain of pasting packets associated with the
1 !       given Display Control Block, updating each.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       ret_status.wlc.v = SMG$$RECALC_PP_FIELDS ( DCB.rab.r )
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       DCB.rab.r       Address of a virtual display control block.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       None
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       None
1 !
1 ! COMPLETION STATUS:
1 !
1 !       SS$_NORMAL      Normal successful completion
1 !       Statuses returned by SMG$$CHECK_OCCLUSION and
1 !                            SMG$$CALC_PASTE_TRANF
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !--
2     BEGIN
2     LOCAL
2         PP : REF $PP_DECL;              ! Addr. of a pasting packet
2
2 !+
2 ! Step through all associated pasting packets, updating each.
2 !-
2     PP = .DCB [DCB_A_PP_NEXT];  ! get 1st packet in DCB-oriented chain
2     WHILE .PP NEQ DCB [DCB_A_PP_NEXT]   ! While any remain...
2     DO
3         BEGIN
3         LOCAL
3             STATUS,                     ! Status of subroutine calls
3             PBCB : REF $PBCB_DECL;      ! Addr of pasteboard control blk
3
3         PBCB = .PP [PP_A_PBCB_ADDR];
3         !+
3         ! Calculate who occludes who in current pasting chain.
3         !-
```

```
; 5928          6154  4             IF NOT (STATUS =SMG$$CHECK_OCCLUSION ( .PBCB))  ! Recalc. occlusion
; 5929          6155  3             THEN
; 5930          6156  3                 RETURN (.STATUS);
; 5931          6157  3
; 5932          6158  3             !+
; 5933          6159  3             ! Calculate critical constants used to map virtual displays
; 5934          6160  3             ! to their correct position within the pasteboard buffer.
; 5935          6161  3             !-
; 5936          6162  4             IF NOT ( STATUS = SMG$$CALC_PASTE_TRANSF ( .PP)) ! Clean up packet
; 5937          6163  3             THEN
; 5938          6164  3                 RETURN (.STATUS);
; 5939          6165  3
; 5940          6166  3             PP = .PP [PP_A_NEXT_DCB];         ! Step to next packet
; 5941          6167  2             END;
; 5942          6168  2
; 5943          6169  2         RETURN ( SS$_NORMAL);
; 5944          6170  1         END;                        ! Routine SMG$$RECALC_PP_FIELDS



                                       000C 00000            .ENTRY   SMG$$RECALC_PP_FIELDS, Save R2,R3                   ; 6098
                      52       04   AC  D0 00002             MOVL     DCB, R2                                             ; 6142
                      53       20   A2  D0 00006             MOVL     32(R2), PP
                      50       20   A2  9E 0000A 1$:         MOVAB    32(R2), R0                                          ; 6143
                      50            53  D1 0000E             CMPL     PP, R0
                                    1D  13 00011             BEQL     2$
                      50       14   A3  D0 00013             MOVL     20(PP), PBCB                                        ; 6150
                                    50  DD 00017             PUSHL    PBCB                                                ; 6154
                 F885 CF            01  FB 00019             CALLS    #1, SMG$$CHECK_OCCLUSION
                      12            50  E9 0001E             BLBC     STATUS, 3$
                                    53  DD 00021             PUSHL    PP                                                  ; 6162
                 F58C CF            01  FB 00023             CALLS    #1, SMG$$CALC_PASTE_TRANSF
                      08            50  E9 00028             BLBC     STATUS, 3$
                      53            63  D0 0002B             MOVL     (PP), PP                                            ; 6166
                                    DA  11 0002E             BRB      1$                                                  ; 6143
                      50            01  D0 00030 2$:         MOVL     #1, R0                                              ; 6169
                                        04 00033 3$:         RET                                                          ; 6170

; Routine Size:  52 bytes,    Routine Base:  _SMG$CODE + 20B8


; 5945          6171  1 !<BLF/PAGE>
```

```
1 %SBTTL 'SMG$$UNPASTE_VIRTUAL_DISPLAY - Unpaste virtual display from pasteboard'
1 GLOBAL ROUTINE SMG$$UNPASTE_VIRTUAL_DISPLAY (
1                                                   DCB :  REF $DCB_DECL,
1                                                   PBCB : REF $PBCB_DECL
1                                               ) =
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       The specified virtual display is "unpasted" from a pasteboard
1 !       if a pasting packet can be found.
1 !       This is the inner-most unpasting routine.  It assumes both
1 !       parameters are present and that they are valid (not necessarily
1 !       that the pasting packet which joins these two exists).
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       ret_status.wlc.v = SMG$$UNPASTE_VIRTUAL_DISPLAY (
1 !                                               DCB.rab.r,
1 !                                               PBCB.rab.r)
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       DCB.rab.r               Address of DCB of virtual display to be
1 !                               unpasted.
1 !
1 !       PBCB.rab.r              Address of the pasteboard from
1 !                               which the unpasting is to take place.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       None
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       None
1 !
1 ! COMPLETION STATUS:
1 !
1 !       SS$_NORMAL      Normal successful completion
1 !       SMG$_NOTPASTED  Specified virtual display is not currently
1 !                       pasted to the specified pasteboard.
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !--
2   BEGIN
2   LOCAL
2       STATUS,                         ! Status of subroutine call
2       PP      : REF $PP_DECL;         ! Addr of pasting packet being
2                                       ! inspected.
2
2 !+
2 ! Try to find the pasting packet joining this DCB and PBCB.
2 ! Exit with SMG$_NOTPASTED if we can't.
2 !-
2
```

```
; 6004      6229 3       IF NOT (STATUS = SMG$$LOCATE_PP ( .DCB, .PBCB, PP))
; 6005      6230 2       THEN
; 6006      6231 2           RETURN (.STATUS);    ! No common pasting packet exists
; 6007      6232 2
; 6008      6233 2   !+
; 6009      6234 2   ! Located desired packet.  Remove it from both queues.
; 6010      6235 2   !-
; 6011      6236 2       $SMG$REMOVE_FROM_QUEUE ( PP [PP_A_NEXT_DCB] );
; 6012      6237 2       $SMG$REMOVE_FROM_QUEUE ( PP [PP_A_NEXT_PBCB] );
; 6013      6238 2
; 6014      6239 2   !+
; 6015      6240 2   ! Give back the pasting packet space
; 6016      6241 2   !-
; 6017      6242 3       IF NOT (STATUS = LIB$FREE_VM ( %REF(PP_K_SIZE), PP))
; 6018      6243 2       THEN
; 6019      6244 2           RETURN (.STATUS);
; 6020      6245 2
; 6021      6246 2   !+
; 6022      6247 2   ! If other virtual displays are still pasted to this pasteboard, we need
; 6023      6248 2   ! to recalculate their occlusion bits since they may have changed by
; 6024      6249 2   ! removing this virtual display.
; 6025      6250 2   !-
; 6026      6251 2       IF .PBCB [ PBCB_A_PP_NEXT] NEQ PBCB [ PBCB_A_PP_NEXT]
; 6027      6252 2       THEN
; 6028      6253 3           IF NOT ( STATUS = SMG$$CHECK_OCCLUSION ( .PBCB ))
; 6029      6254 2           THEN
; 6030      6255 2               RETURN (.STATUS);
; 6031      6256 2
; 6032      6257 2   !+
; 6033      6258 2   ! Cause pasteboard to reflect this change.
; 6034      6259 2   !-
; 6035      6260 2
; 6036      6261 2      RETURN ( SMG$$CHECK_FOR_OUTPUT_PBCB ( .PBCB ));
; 6037      6262 2
; 6038      6263 1      END;                            ! Routine SMG$$UNPASTE_VIRTUAL_DISPLAY



                                     0004 00000        .ENTRY  SMG$$UNPASTE_VIRTUAL_DISPLAY, Save R2         ; 6173
                 5E               08  C2 00002        SUBL2   #8, SP
                           04     AE  9F 00005        PUSHAB  PP                                            ; 6229
                 7E        04     AC  7D 00008        MOVQ    DCB, -(SP)                                    ;
          FE71   CF               03  FB 0000C        CALLS   #3, SMG$$LOCATE_PP                            ;
                 3C               50  E9 00011        BLBC    STATUS, 2$                                    ;
                 51        04     BE  0F 00014        REMQUE  @PP, FOO                                      ; 6236
   51      04    AE               08  C1 00018        ADDL3   #8, PP, R1                                    ; 6237
                 52               61  0F 0001D        REMQUE  (R1), FOO                                     ;
                           04     AE  9F 00020        PUSHAB  PP                                            ; 6242
           04    AE               37  D0 00023        MOVL    #55, 4(SP)                                    ;
                           04     AE  9F 00027        PUSHAB  4(SP)                                         ;
     00000000G   00               02  FB 0002A        CALLS   #2, LIB$FREE_VM                               ;
                 1C               50  E9 00031        BLBC    STATUS, 2$                                    ;
           08    AC        08     BC  D1 00034        CMPL    @PBCB, PBCB                                   ; 6251
                                  0B  13 00039        BEQL    1$                                            ;
                           08     AC  DD 0003B        PUSHL   PBCB                                          ; 6253
```

```
                        FB2C   CF               01  FB 0003E         CALLS   #1, SMG$$CHECK_OCCLUSION                            ;
                               0A               50  E9 00043         BLBC    STATUS, 2$                                          ;
                                        08      AC  DD 00046 1$:     PUSHL   PBCB                                                ; 6261
                   00000000G   00               01  FB 00049         CALLS   #1, SMG$$CHECK_FOR_OUTPUT_PBCB                      ;
                                                    04 00050 2$:     RET                                                         ; 6263

; Routine Size:  81 bytes,    Routine Base:  _SMG$CODE + 20EC


; 6039          6264  1 !<BLF/PAGE>
```

```
                                                                E  9
SMG$DISPLAY_LIN SMG$DISPLAY_LINKS - Virtual Display Linkages    16-Sep-1984 00:29:22    VAX-11 Bliss-32 V4.0-742             Page 176
1-096           SMG$$UNPASTE_VIRTUAL_DISPLAY - Unpaste virtual  14-Sep-1984 13:09:43    [SMGRTL.SRC]SMGDISLIN.B32;1              (38)

; 6041          6265  1 END                         ! End of module SMG$DISPLAY_LINKS
; 6042          6266  1
; 6043          6267  0 ELUDOM



                                                                        .EXTRN  LIB$SIGNAL

;                                    PSECT SUMMARY
;
;       Name                       Bytes                            Attributes
;
;   _SMG$DATA                          80  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;   _SMG$CODE                        8509  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)



;                               Library Statistics
;
;                                            -------- Symbols --------        Pages       Processing
;       File                                 Total     Loaded   Percent       Mapped      Time
;
;   _$255$DUA28:[SYSLIB]STARLET.L32;1         9776        101         1          581       00:00.9
;   _$255$DUA28:[SMGRTL.OBJ]RTLLIB.L32;1        36          0         0            8       00:00.1
;   _$255$DUA28:[SMGRTL.OBJ]SMGLIB.L32;1       469        152        32           38       00:00.4



;                                    COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS$:SMGDISLIN/OBJ=OBJ$:SMGDISLIN MSRC$:SMGDISLIN/UPDATE=(ENH$:SMGDISLIN
;       )

; Size:           8471 code + 118 data bytes
; Run Time:          02:45.3
; Elapsed Time:      08:00.6
; Lines/CPU Min:      2275
; Lexemes/CPU-Min: 20356
; Memory Used:  435 pages
; Compilation Complete
```

0356 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

SMGDISDRW
LIS

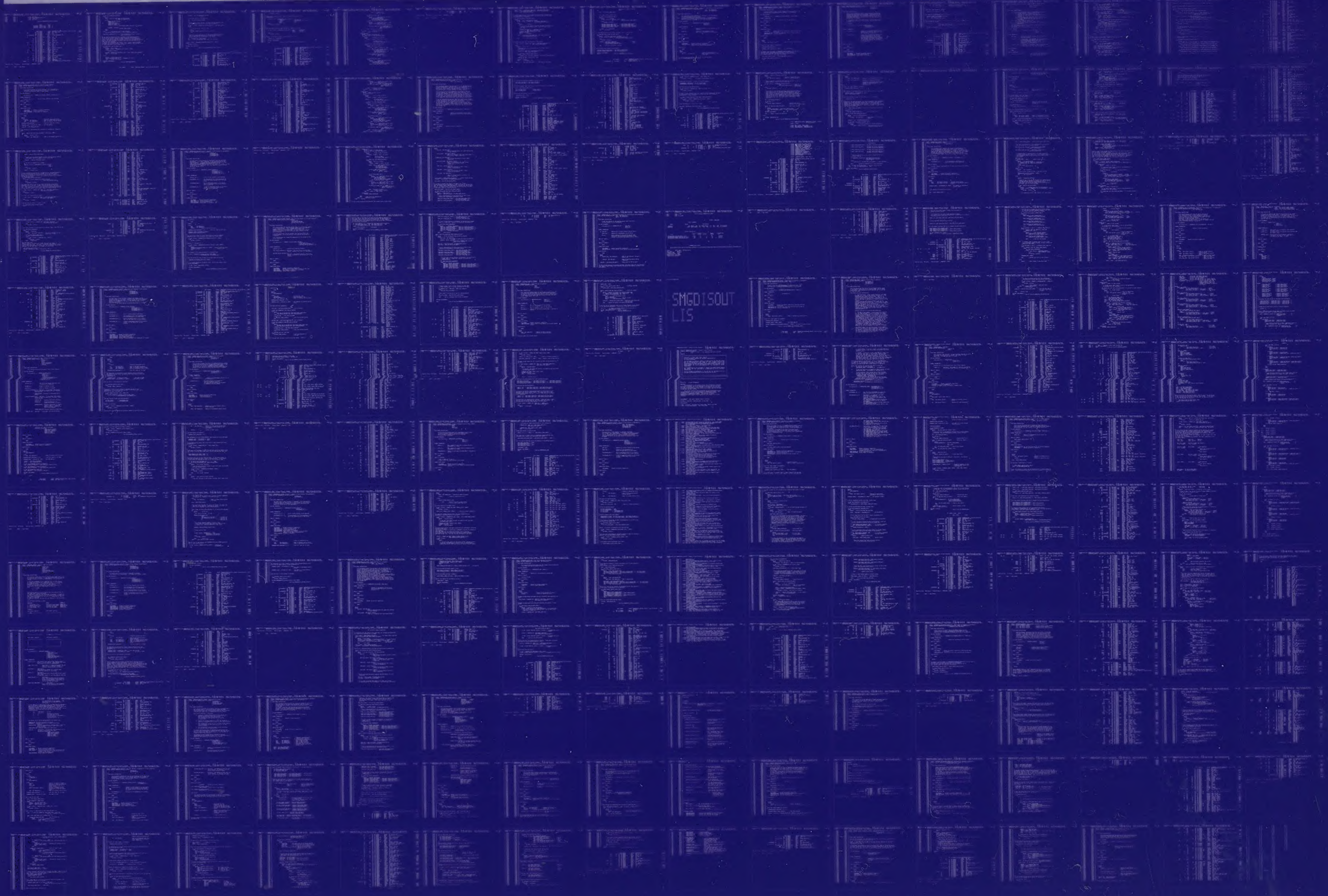
SMGDISLIN
LIS

SMGDISINP
LIS

SMGDISDHW
LIS

0357 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY


SMGDISOUT
LIS